AMS ADVANTAGE ® Pervasive Data Integrator Training

Presentation Overview

- Introductions
- Course Objectives
- Introduction to Data Integrator
- Conversion and Interface Overview
- DI Map Transformation Examples
- Questions

Introductions

- Name
- Project
- Experience with Data Integrator

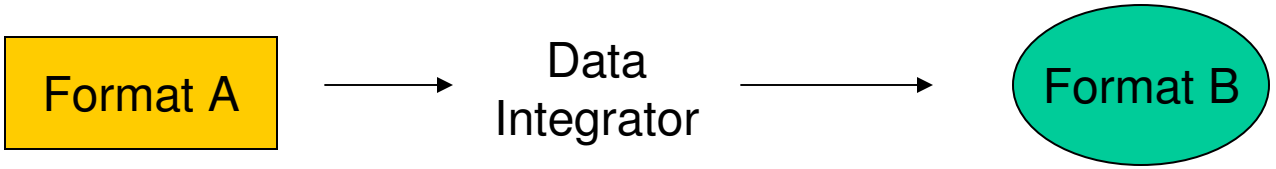
Objectives for the Course

- Acquire basic understanding of Data Integrator
  - Familiarity with tools
  - Various options
- Introduction to the process of using DI with the Advantage products
  - Advantage specific uses – documents, tables
  - Advantage examples

Introduction to Pervasive Data Integrator

Data Integrator

- Data Integrator (DI) is a tool used to convert data from one format to another



- Flexible to allow for many input and output data formats
- Use DI to convert only parts of the data

Benefits of using Data Integrator®

- Data Integrator® Studio functions as a flexible code generator, using rules defined by the developer.
- Supports look ups to tables either within the ADVANTAGE® RDBMS data structure or in the staging / conversion database.
- Makes use of reusable code modules.
- The Rapid Integration Flow Language (RIFL) encompasses functions, statements and keywords that are used in Source/Target Filters, Target Field Expressions, and Code Modules. RIFL mimics VBScript and Visual Basic functions.

File Limitations of Data Integrator®

- Limited to one input file at a time.
  - This one file will be the driver file.
  - All additional information can be loaded into tables and accessed from the Data Integrator® code.
- Only two output files at a time for performance reasons
  - Processed records and Rejected records
  - Generate multiple outputs with multiple program passes.
  - Log File available for error messages.

CGI-AMS DI Usage

- AMS Advantage conversion process uses DI to convert data from 2x to 3x
- Table to Table conversions and document conversions and interfaces
- 3x application accepts data that is in a specified XML format
- DI takes input data and produces an XML file for table to table conversions, document conversions and in-bound interfaces
- DI takes XML and produces an output file for out-bound interfaces

DI License Issue

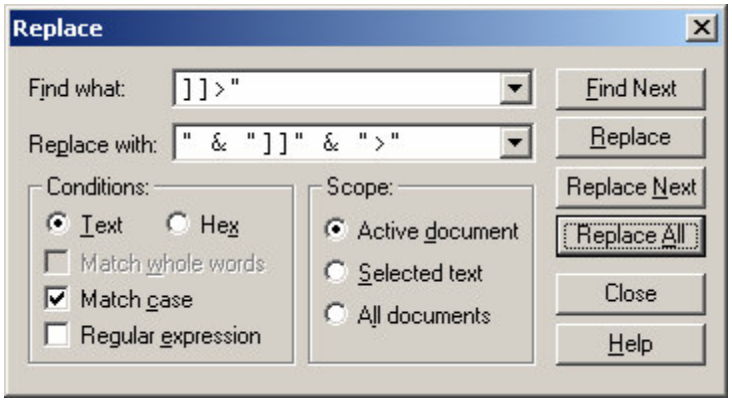
- Two Separate License files for Studio and Engine
- Open Map Designer
  (Help -> About Map Designer -> License)
- C:\winnt\DI800.ini
- C:\winnt\DI800_MAP.ini
- C:\winnt\DI800_ENGINE.ini
- Ask Pervasive for one license file

Issues with DI 8.x

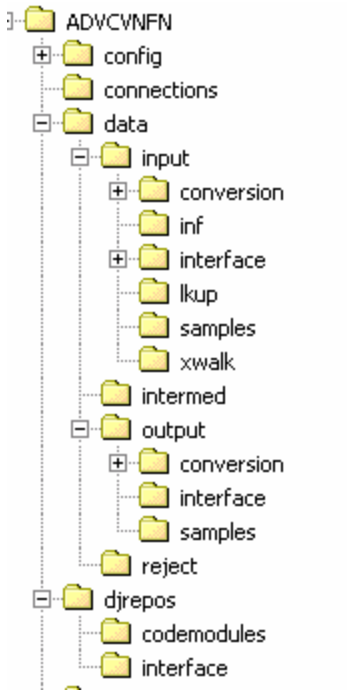
- If you are running DI 8.4 * you will need to run the maintenance patch to fix issues with the baseline conversion software for HR and FIN.
- Patch is available through Pervasive
- DI map designer in 8.x initializes fields differently than DI engine.
- One sets them to ‘null’ and the other sets it to ‘0’. Baseline code handles it by checking for both the conditions.
- If the patch is not installed, there will be errors when running the maps
- ftp://polar.datajunction.com/downloads/840/Win32/Maint/Maint8417_20041019ER.exe

Issues with DI 8.x Contd …

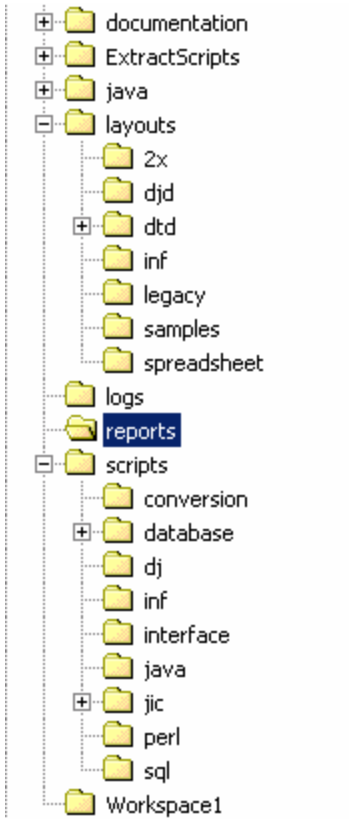
- Open the xml file using Textpad and replace]]>” with “ & “]]” & “>”



Directory Structure



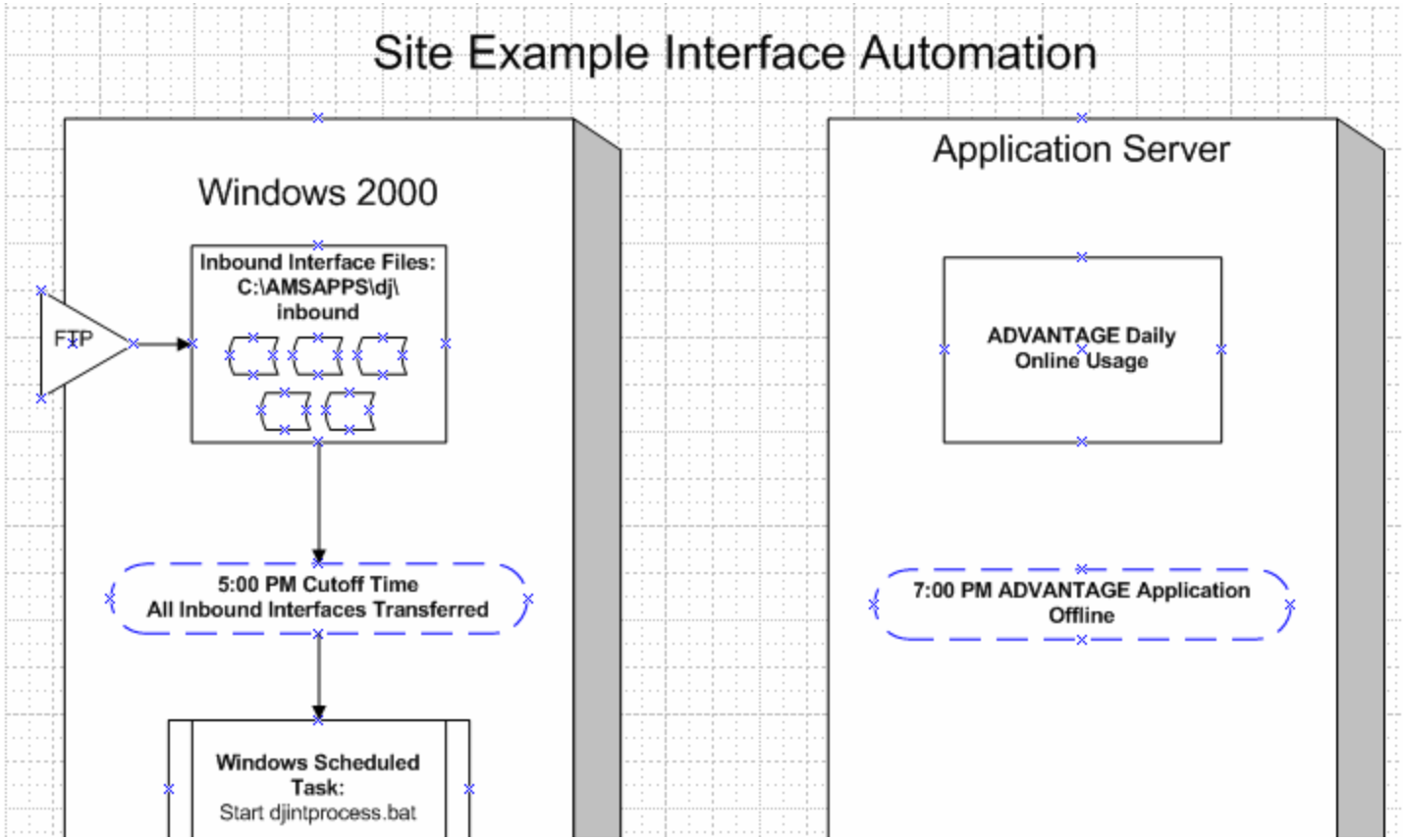
Directory Structure Contd …



Conversion File Structure

- **CONFIG -** contains .ini files and other support files needed to process the conversion.
- **DATA -** contains the data being pulled into or being pushed out from conversion.
- **INPUT -** holds the files used as input to PDI that were created from the data extracted from the ADVANTAGE FIN 2.2 system.
- **INTERMED -** The ADVANTAGE Financial conversion process does not use this folder
- **OUTPUT -** holds the output files after the PDI process has transformed them.
- **REJECT -** The ADVANTAGE HR conversion process does not use this folder
- **DJREPOS -** contains the PDI internal database, along with the individual maps used as stand-alone items.
- **DOCUMENTS -** contains documentation about the ADVANTAGE FIN 2.2 to ADVANTAGE FIN 3.5 conversion. It includes a master listing of all of the ADVANTAGE FIN 2.2 tables and ADVANTAGE FIN 3.5 tables that are part of conversion.
- **EXTRACTSCRIPTS -** contains all of the scripts needed to extract the data from the ADVANTAGE FIN 2.2 database. For MVS sites, this includes the JCL in their respective JOBLIB, PROCLIB, and PARMCARD folders. For Unix sites, this includes the C-shell scripts and needed parameters in the PARM folder.
- **JAVA -** contains all of the Java code to run the Java Update conversion scripts. The supporting software for running the scripts are located in the ADVANTAGE, COM, CONFIG, JAVADOC, UTILS folders and sub folders. The Java scripts themselves are located in the UTILS folder.
- **LAYOUTS -** contains COBOL layouts that where used to assist in setting up the input file layout. This folder has been included for reference purposes only.
- **LOGS –** contains log files that are created through out the conversion process.
- **SCRIPTS -** contains all of the scripts needed to transform the ADVANTAGE Fin 2.x data in to its corresponding Advantage 3.x format.

Site Specific Process



Install Folders

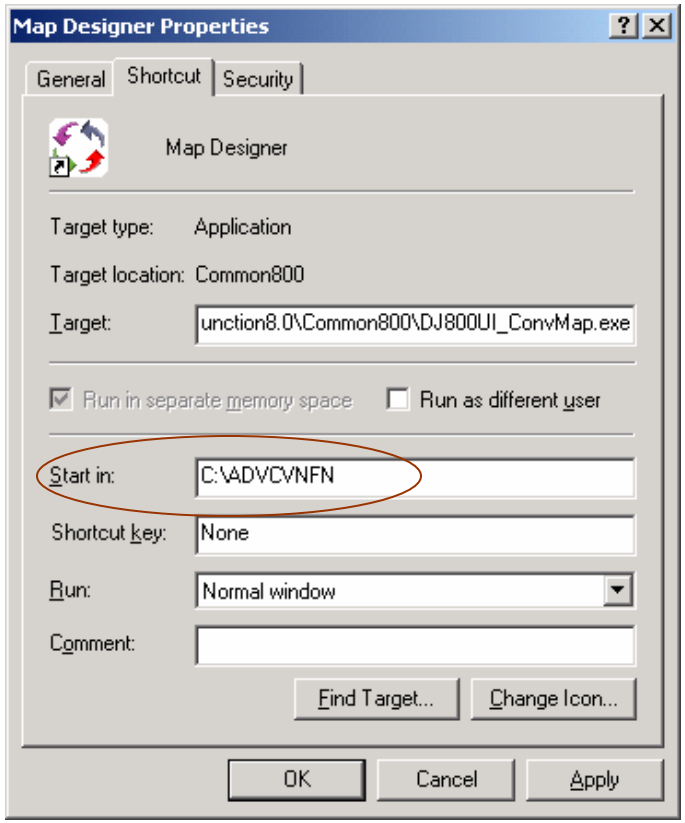
- C:\ADVCVNFN
- C:\ADVCVNHR

Installation Procedure

- Install DataJunction 8.x
- Install the Patch
- Extract ADVCVNFN.zip onto C:\

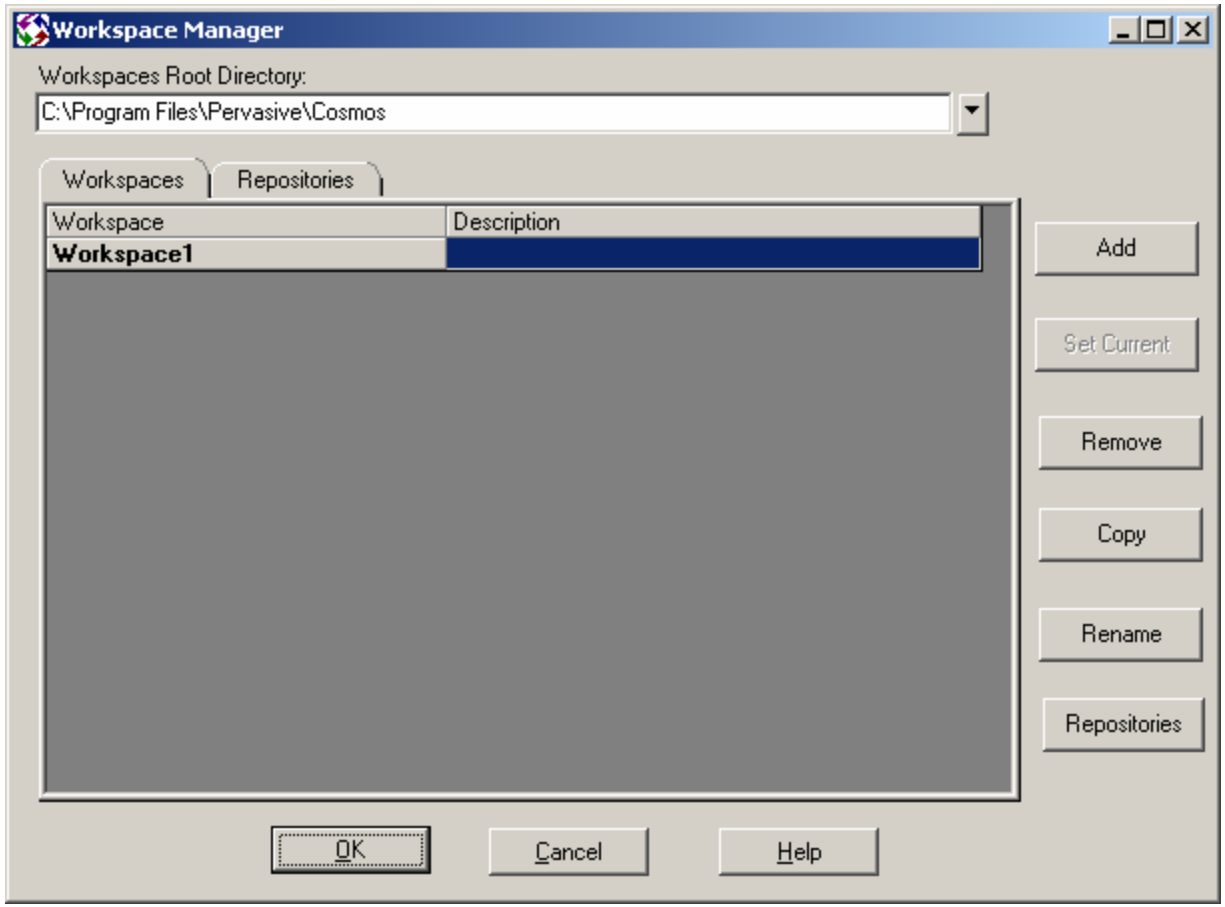
Installation Procedure

- Modify the Map Designer Properties “Start in” path to use relative paths. Set to C:\ADVCVNFN.



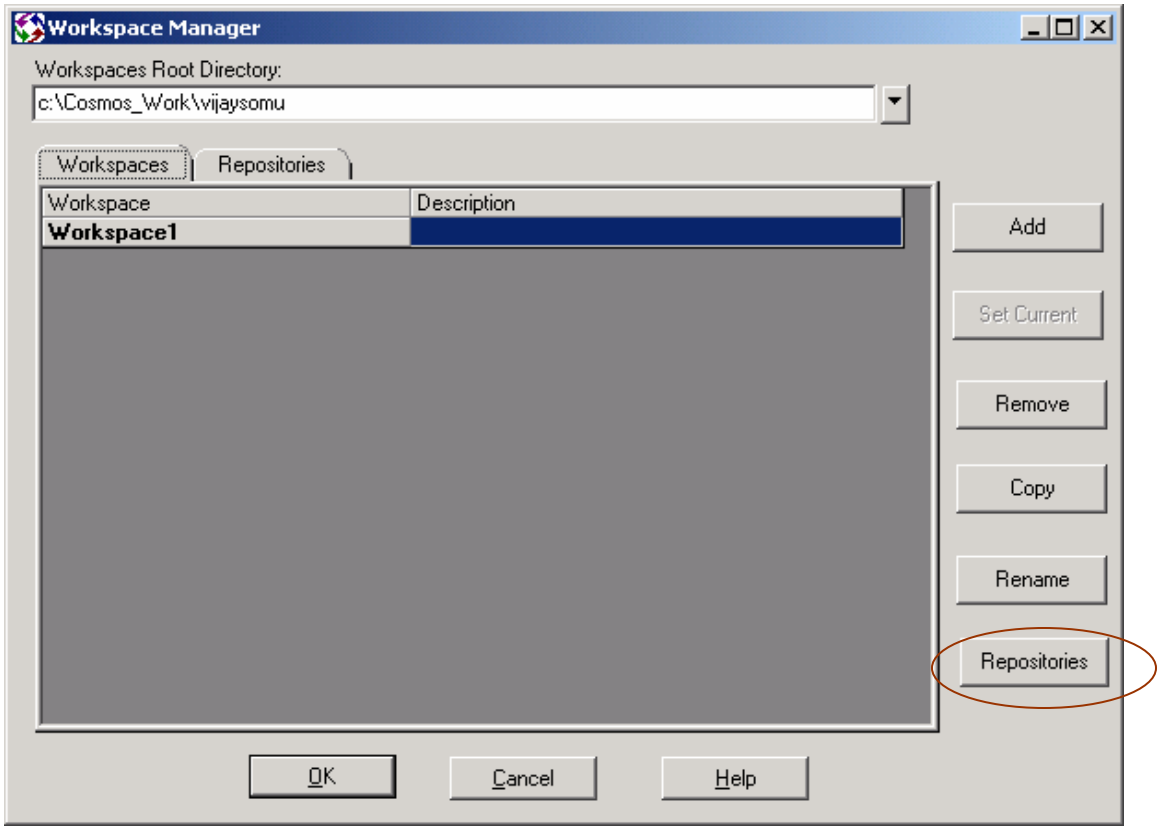
Installation Procedure

- Open the Map Designer and select File → Manage Workspaces…



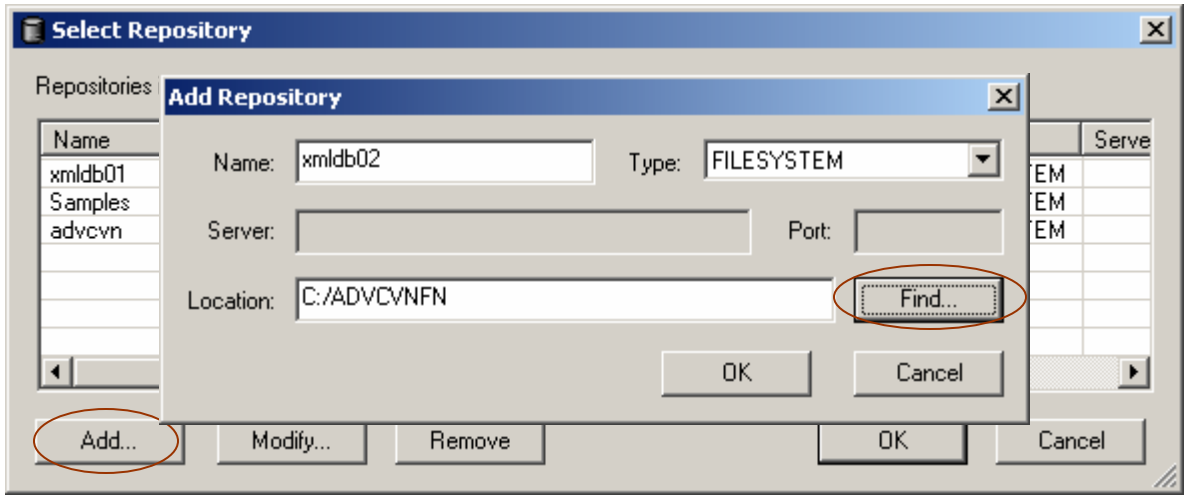
Installation Procedure

- Click on Repositories



Installation Procedure

- Click on Add and then Find to add the C:/ADVCVNFN repository



Installation Procedure

- Install Oracle Enterprise (if needed – else install Oracle Client)
- Install DB2 Client (in required by your site)
- Create an ODBC Connection to the conversion database

Installation Procedure Contd …

- Create a DI Map designer shortcut
- Modify the setenv.bat script in the ADVCVNFN folder
- Modify the conv301.ini file in the ADVCVNFN\config folder
- Modify the script SetEnvironVar.bat in the ADVCVNFN\scripts\inf folder

Conv301.ini

- Configuration file that contains database and java path information. Need to modify to point to correct paths.
  - JAVA_CLASS_PATH : Full path names to files
  - JAVA_LOG_FILE : Full path names to file
  - CONV_DB_CONN : Desire to run software with the support of a conversion database
  - APP_DB_CONN : Desire to run software with the support of an application database
  - CONV_DB_TYPE : Type of Conversion database present
  - APP_DB_TYPE : Type of Application database present
  - CONV_ORACLE : Specific connection info for an Oracle database connection
  - CONV_DB2 : Specific connection info for a DB2 database connection
  - APP_ORACLE : Specific connection info for an Oracle database connection
  - APP_DB2 : Specific connection info for a DB2 database connection

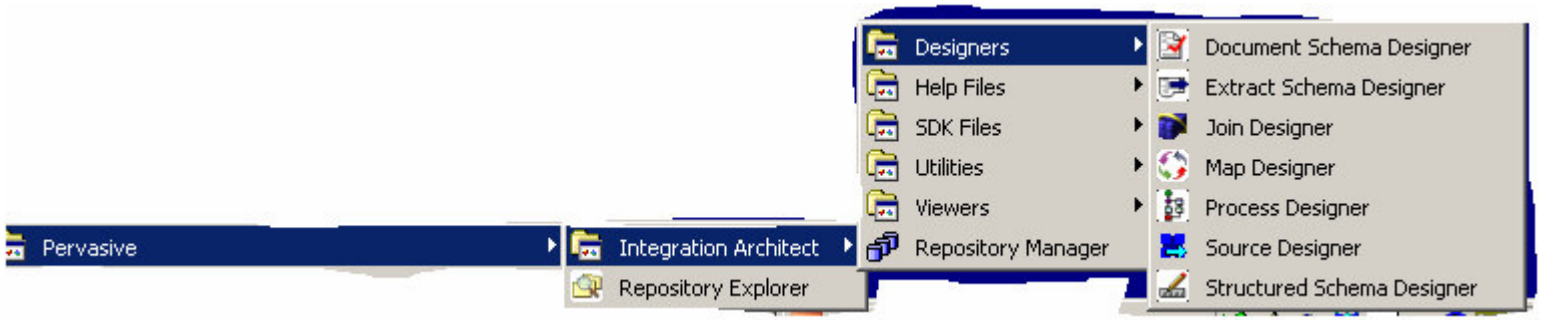
Setbat.env

- File that sets the environment to allow bat scripts to be run from any directory
- Sets parameters established in conv301.ini

Data Integrator Overview

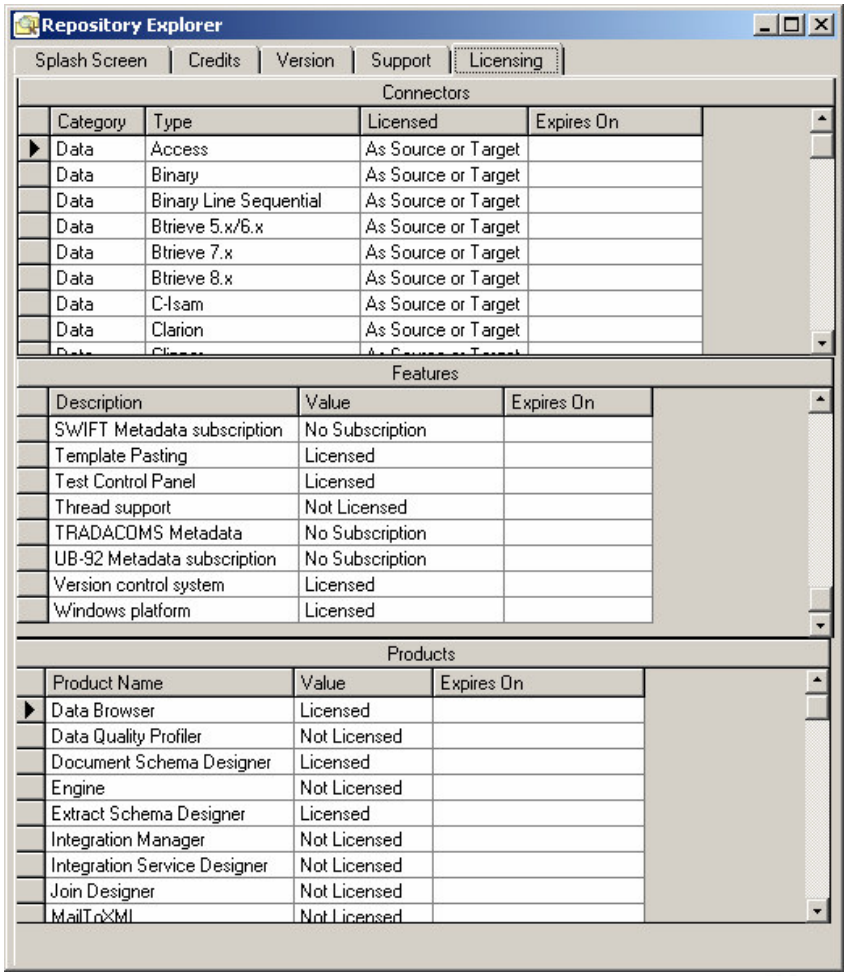
Data Integrator Overview

- Different Parts
  - Repository Explorer
  - Integration Architect
    - Map Designer (Studio)
    - Integration Engine



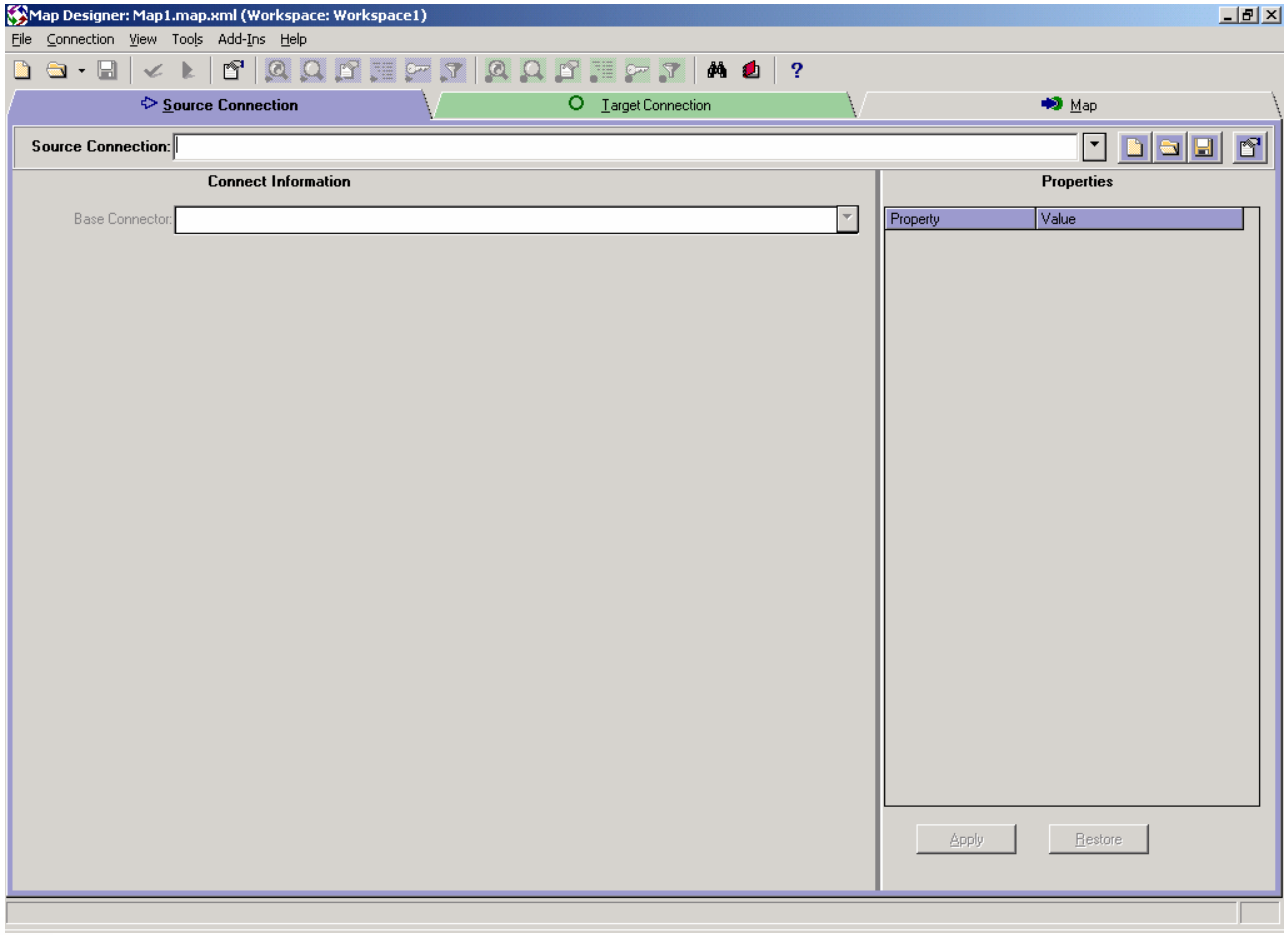
Repository Explorer

- Navigate the repository
- Navigate to other Integration Architect tools
- Can work with CVS or VSS
- View Licensing information



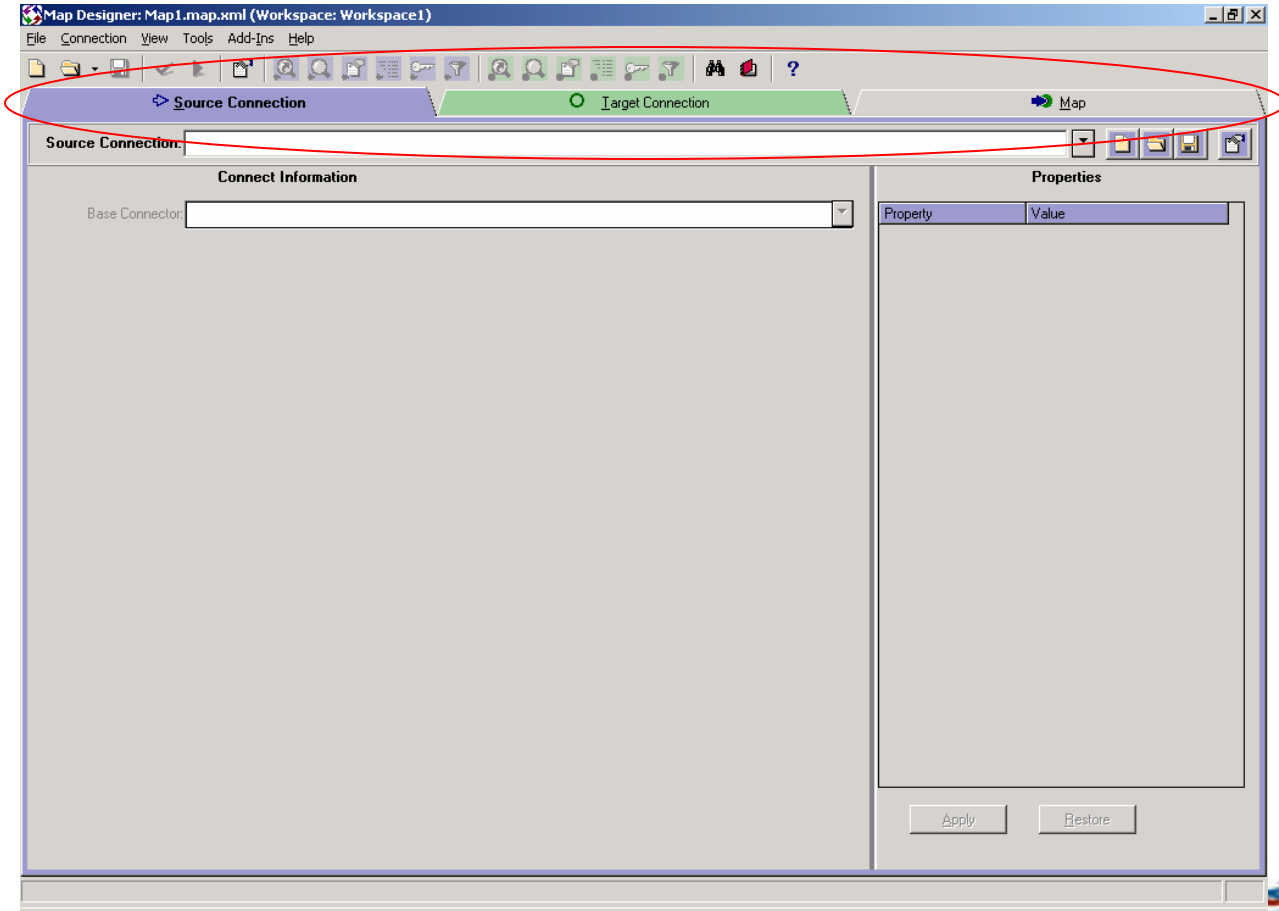
Map Designer

- Construct maps



Map Designer

- Three main tabs
  - Source
  - Target
  - Map

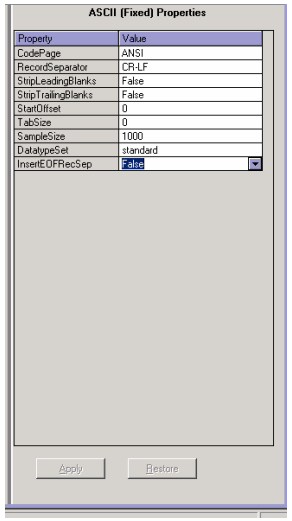


Map Designer - Source

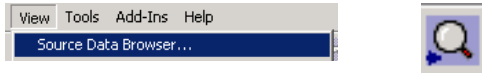
- Source
  - Specifies the “from” of the conversion
  - Many options in source tab
    - Source Type
    - Source File
    - Server Name
    - Source Data Source
    - User ID
    - Password
    - Source Table

Map Designer - Source

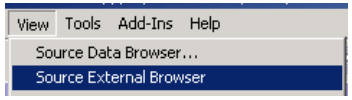
- Multiple source properties
  - Dependent on the type of source chosen



  - View the data to ensure the properties are correct
    - Internal Data Browser



    - External Data Browser

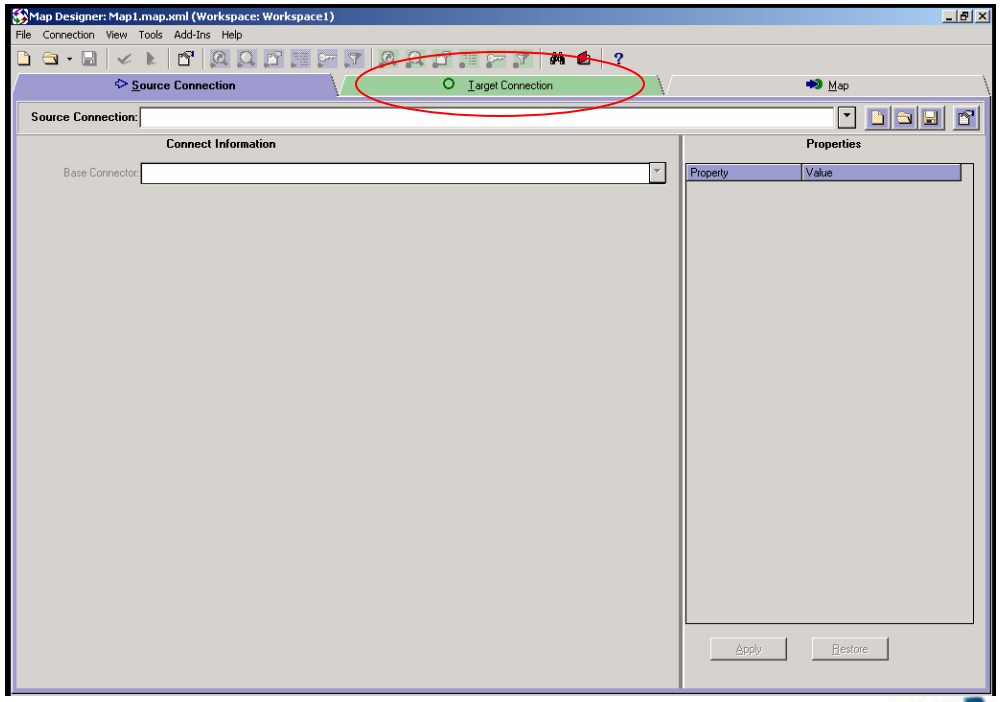


Map Designer - Source

- Source Record Layout
  - Contains file information
    - Length of records and fields
    - Data types of the fields
    - Properties
  - Source Record Types Designer
    - Setup record layout of source file

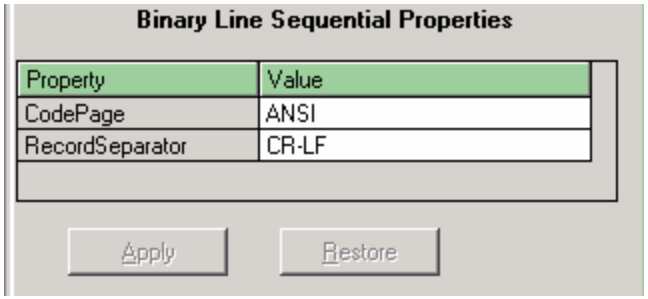
Map Designer - Target

- Specifies the “to” in the conversion
- Need to set several options
  - Target Type
  - Target File or Target Database and Table
  - Output Mode



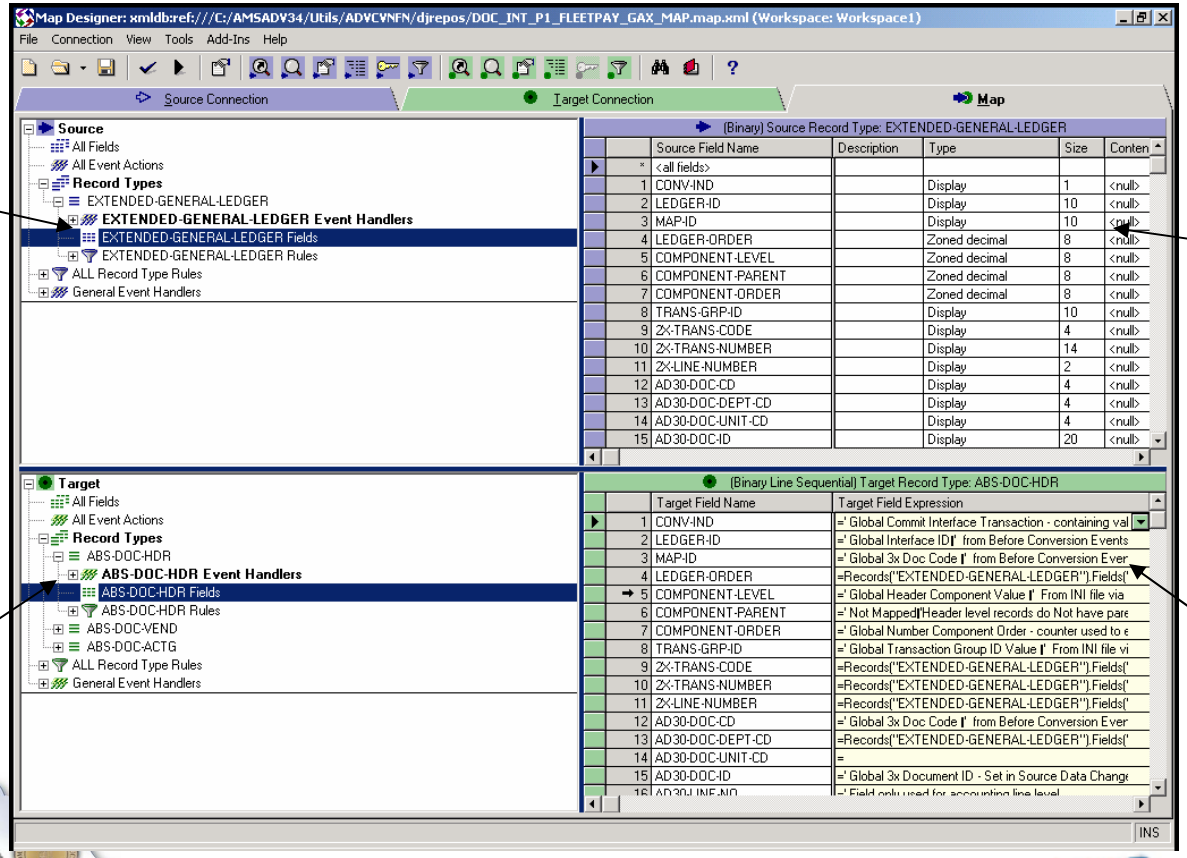
Map Designer - Target

- Target Properties



Map Designer - Map

- Map divided into four sections



Source tree structure

Source grids

Target tree structure

Target grids

Map Designer - Map

- Specify where you want the data to appear in the Target file and the relationships between Source and Target data
- Drag and drop fields from source to match fields in target
- Can create numeric and logical expressions to manipulate the data
- Can select a subset of the total fields to be written to the target file

Map Designer - Map

- Source Tree
  - Displays non-editable information about source connection
  - Navigate through the fields in the recordsource
  - Allows creation of EventHandlers
    - occur during the reading of Source data, and before the transformation and writing of the data.

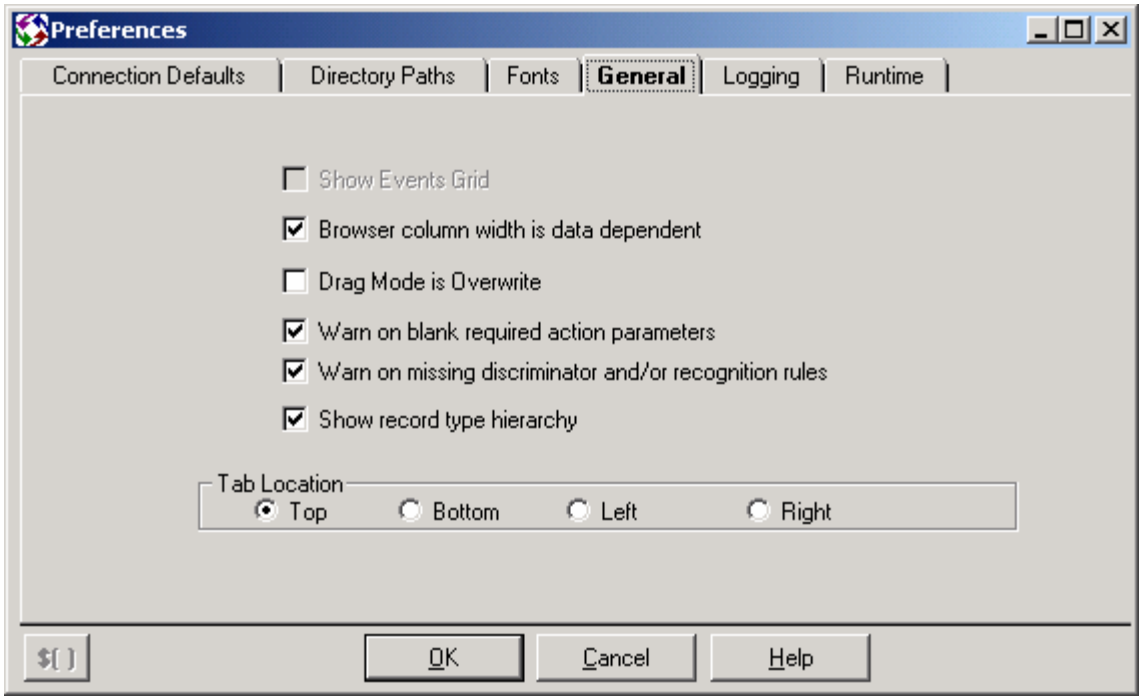
Map Designer – Map

- Target Tree
  - Displays similar information but for target instead of source
  - Properties in the map specify exactly what is written to the target data file
  - Before mapping the target field name and target field expression cells contain no information
  - After mapping target field name and field expression contain the mapped information

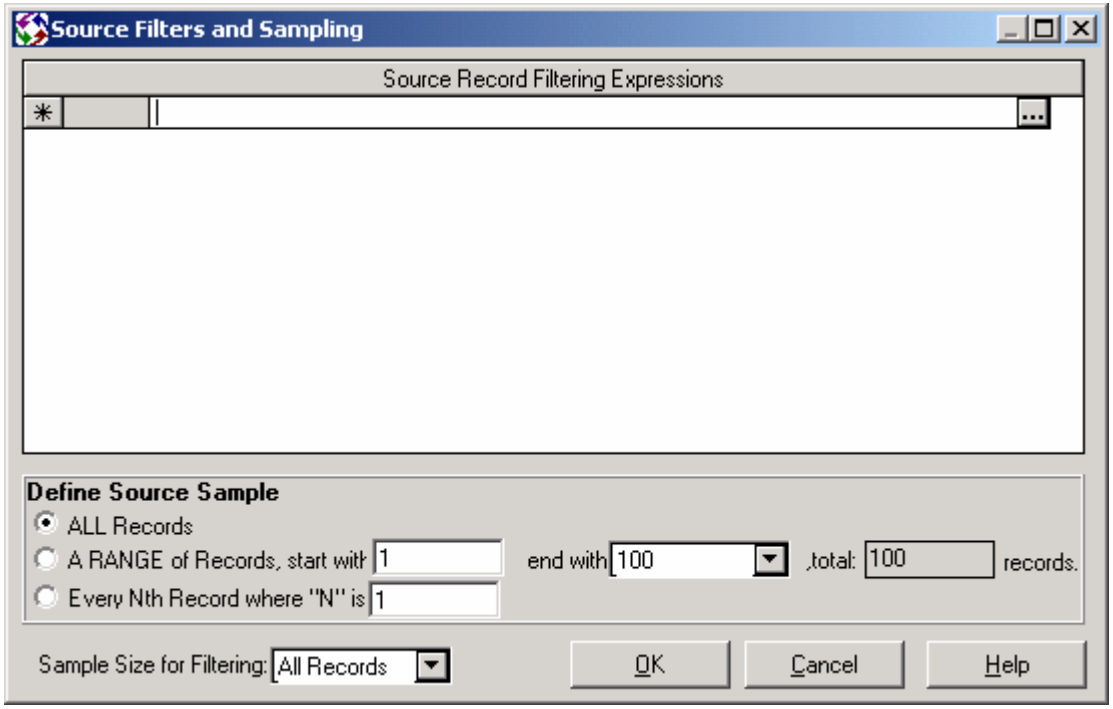
Run a conversion

- After all the target information has been mapped the conversion can be run
  - Save and Validate conversion map 
  - Run conversion 
  - Look at target data 

Properties

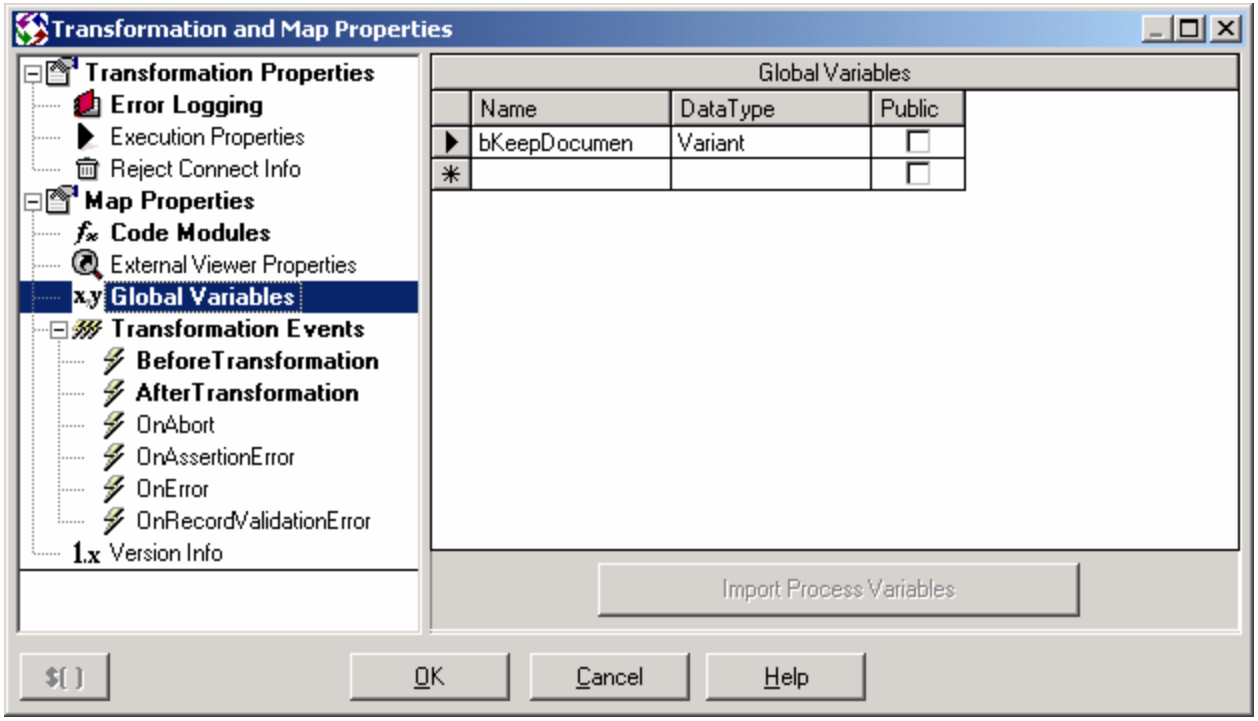

Preferences
Connection Defaults
Directory Paths
Fonts
General
Logging
Runtime
Show Events Grid
Browser column width is data dependent
Drag Mode is Overwrite
Warn on blank required action parameters
Warn on missing discriminator and/or recognition rules
Show record type hierarchy
Tab Location
Top
Bottom
Left
Right
$( )
OK
Cancel
Help

Source and Target Filtering

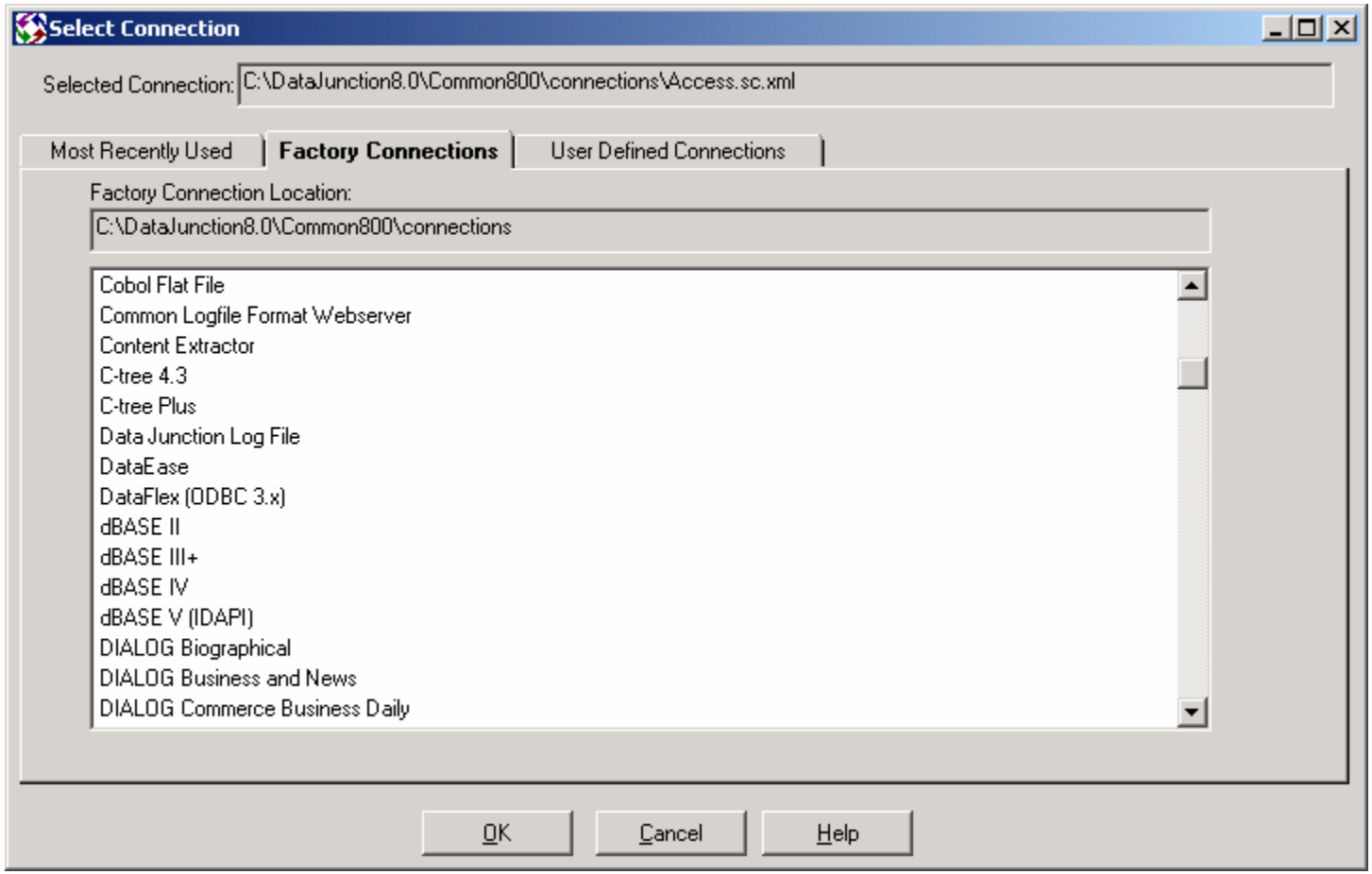


Conversion Properties

- Global Variables



Source and Target Layouts



Target Tree – Field Expression

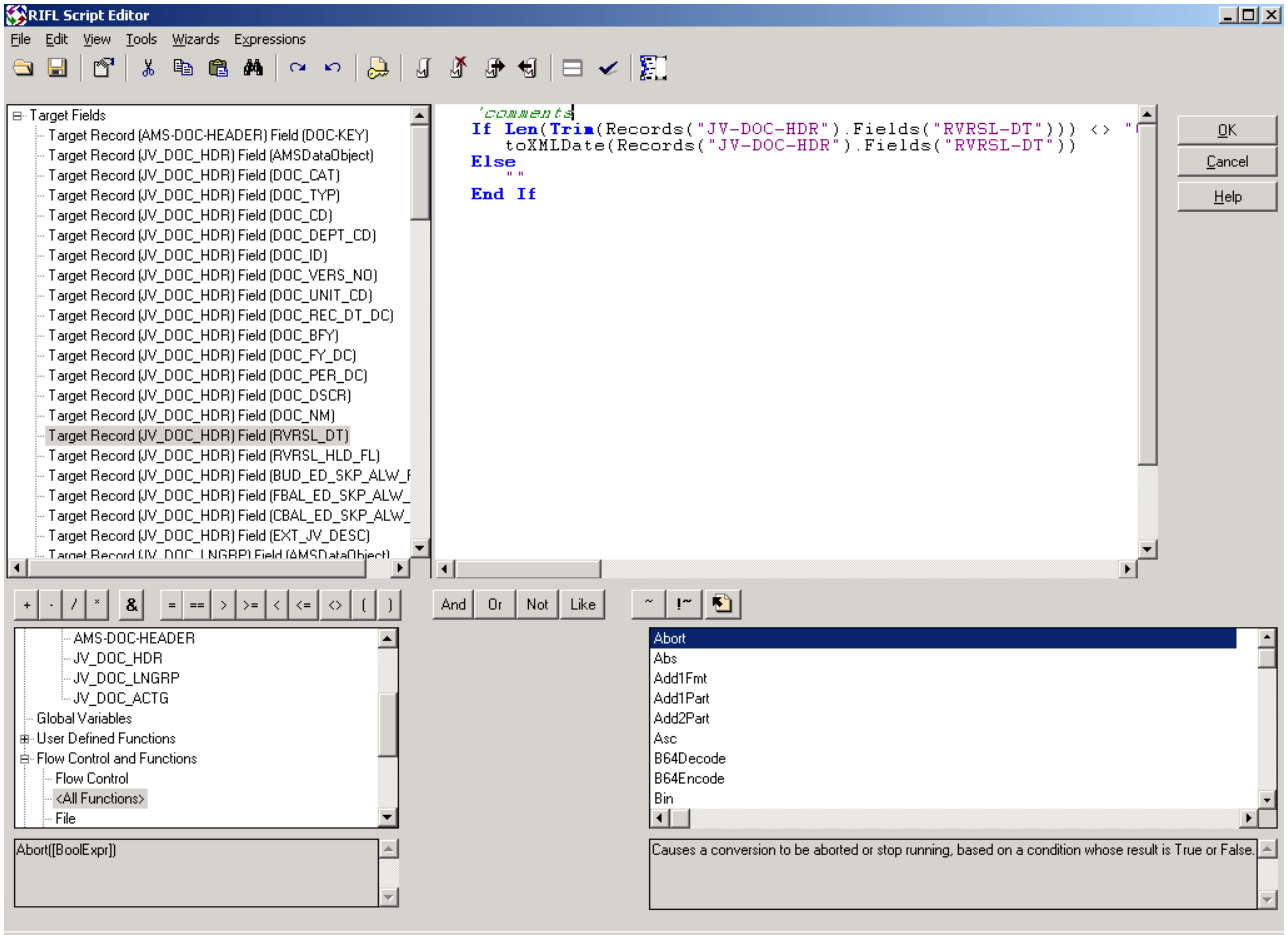
- Click on the arrow to view the Expression Builder

(XML) Target Record Layout: AMS_DOC_XML_EXPORT_FILE

		Target Field Name	Target Field Expression	Description	Type	Size	Results
	1	VERSION	=1		Attribute	16	1
▶	2	EXPORT_DATE	ine= DateConvert("yyyy/mm/dd","yyyy-mm-d		Attribute	16	2004-06-09 17:40:26.000000
	3	AMS_DOCUMENT	=		Record		<null>
*							

Expression Builder

- Allows expressions for fields to be written



Expression Builder

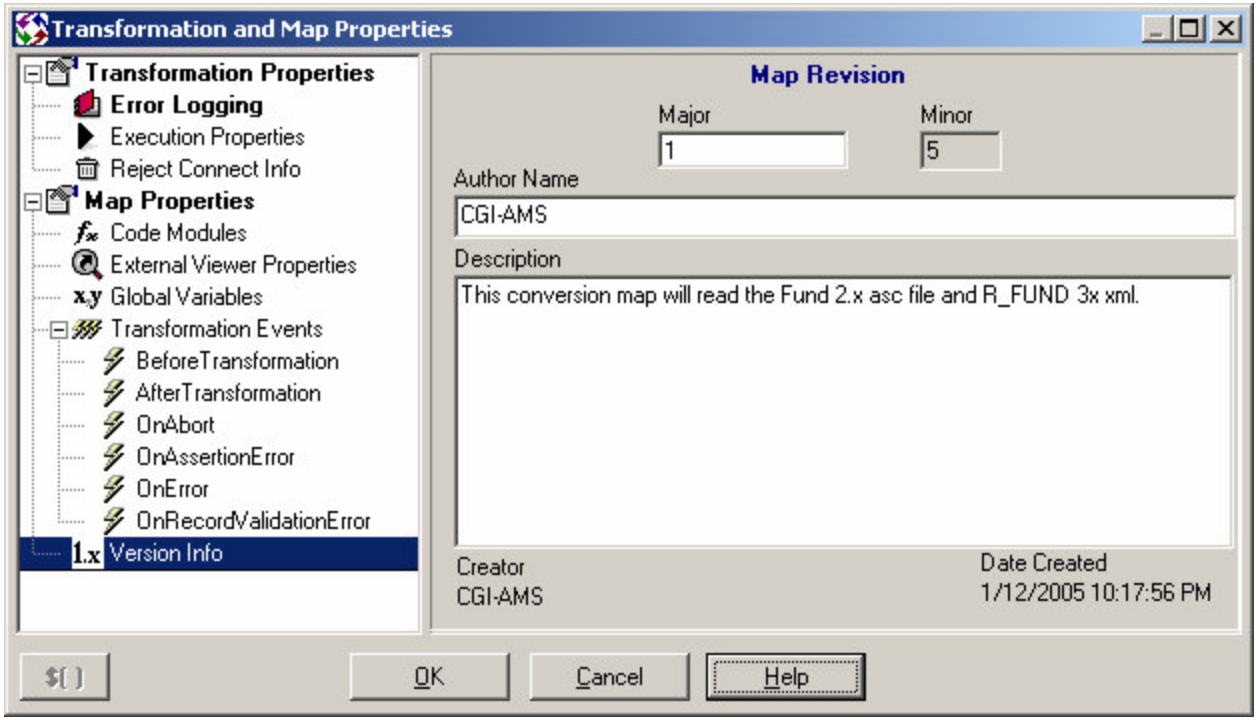
- Validate expressions
- Show expression tree
- Operator buttons
- Field names

Expression Builder

- Font color represents type of information
  - key words are bold Blue
  - default text and background are black,
  - comments are green
  - quoted text is purple

Conversion Properties

- Set Conversion properties 



Conversion Properties

- Set various properties used throughout the conversion
  - Version info – allows traceability of versions of conversions
  - Conversion Events – events related to entire conversion
  - Code Modules – external code modules that can be called from within the conversion

Source Event Precedence

- BeforeTransformation events
- Source Filter Expressions (for each record)
- Source Sort (for each record)
- Source Filter Range (for all records)
- BeforeFirstRecord (Source)
- BeforeEveryRecord (Source)
- BeforeFirstRecord (Source General)
- BeforeEveryRecord (Source General)

Source Event Precedence Contd …

- OnAnyDataChange
- OnAllDataChange
- OnDataChange1-5
- AfterFirstRecord (Source)
- AfterEveryRecord (Source)
- AfterFirstRecord (Source General)
- AfterEveryRecord (Source General)
- On End of File (Source General)
- AfterTransformation events

Target Event Precedence

- Mapping Precedence
  - Target General Before Map event
  - Target Record Before Map event
  - Map Action
  - Target Record After Map event
  - Target General After Map event

Target Event Precedence Contd …

- Put Precedence
  - Target General Before Put Record event
  - Target Record Before Put Record event
  - Put action
  - Target Record After Put Record event
  - Target General After Put Record event

Target Event Precedence Contd …

- Error Precedence
  - Expression-level Error (i.e., On Error Goto)
  - Source Record Error OR Target Record Specific Error
  - Source General Error OR Target General Specific Error
  - Target Record OnError
  - Target General OnError
  - Transformation-level Error

Events

- Event Handlers
  - Actions can be triggered at virtually any point in the conversion process
    - ClearMapPut Record - The Target record buffer is cleared, Target field expressions are executed, and the resulting data is written out to the Target file
    - Execute – Runs an expression

Basic File to File Conversion

- Input file – Title.txt

```
INACT 19500101999999999INACTIVE                          INACTIVE
1000  20030202999999999ADMINISTRATIVE TECHNICIAN I       ADM TECH I
1000  20010107200302O1ADMINISTRATIVE TECHNICIAN I       ADM TECH I
1000  20001001200101O6ADMINISTRATIVE TECHNICIAN I       ADM TECH I
1000  20000109200009O0ADMINISTRATIVE TECHNICIAN I       ADM TECH I
```

- Desired output – Titles.txt

```
INACT 80499898_@_INACT_@_INACTIVE_@_INACTIVE_@__@_N_@_F_
1000  79969797_@_1000_@_ADMINISTRATIVE TECHNICIAN I_@_ADI
1000  79989892_@_1000_@_ADMINISTRATIVE TECHNICIAN I_@_ADI
1000  79998998_@_1000_@_ADMINISTRATIVE TECHNICIAN I_@_ADI
1000  79999890_@_1000_@_ADMINISTRATIVE TECHNICIAN I_@_ADI
1000  80018995_@_1000_@_ADMINISTRATIVE TECH I_@_ADM TECH
1000  80038770_@_1000_@_ADMINISTRATIVE TECH I_@_ADM TECH
1000  80039271_@_1000_@_ADMINISTRATIVE TECH I_@_ADM TECH
```

- Include field separator and removing fields

Basic File to File Conversion

- Input File – House.txt

```
City     5        3        Blue
Suburb   3        1        Red
City     2        1        White
Suburb   4        3        Brick
Country  10       7        Brick
```

- Output File – HouseOut.txt

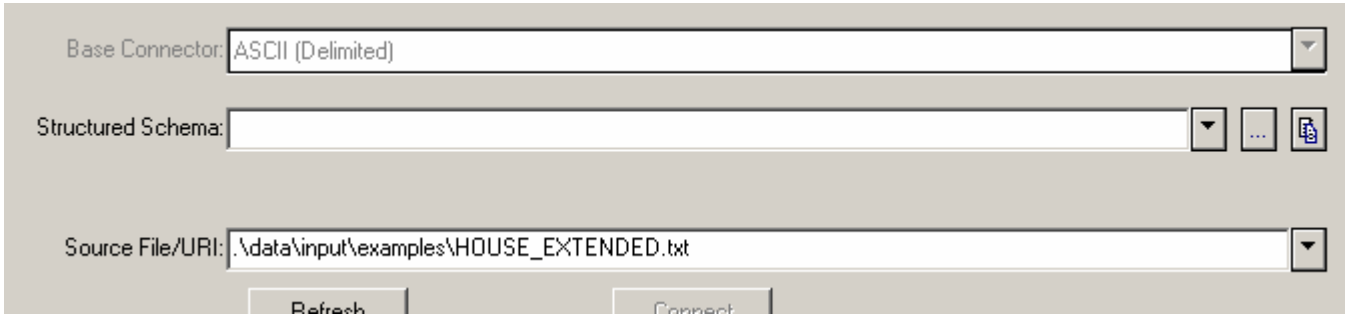
```
City     Blue     5        3        Good
Suburb   Red      3        1        Good
City     White    2        1        Good
Suburb   Brick    4        3        Good
Country  Brick    10       7        Good
```

- Rearrange fields and add a field

Exercise

- Input File – House_Extended.txt

```
Virginia        City    5       3       Blue    Driveway
Florida Suburb  3       1       Red     Golf Course
New York        City    2       1       White   Garage
California      Suburb  4       3       Brick   Lake
Washington      Country 10      7       Brick   Farm
```

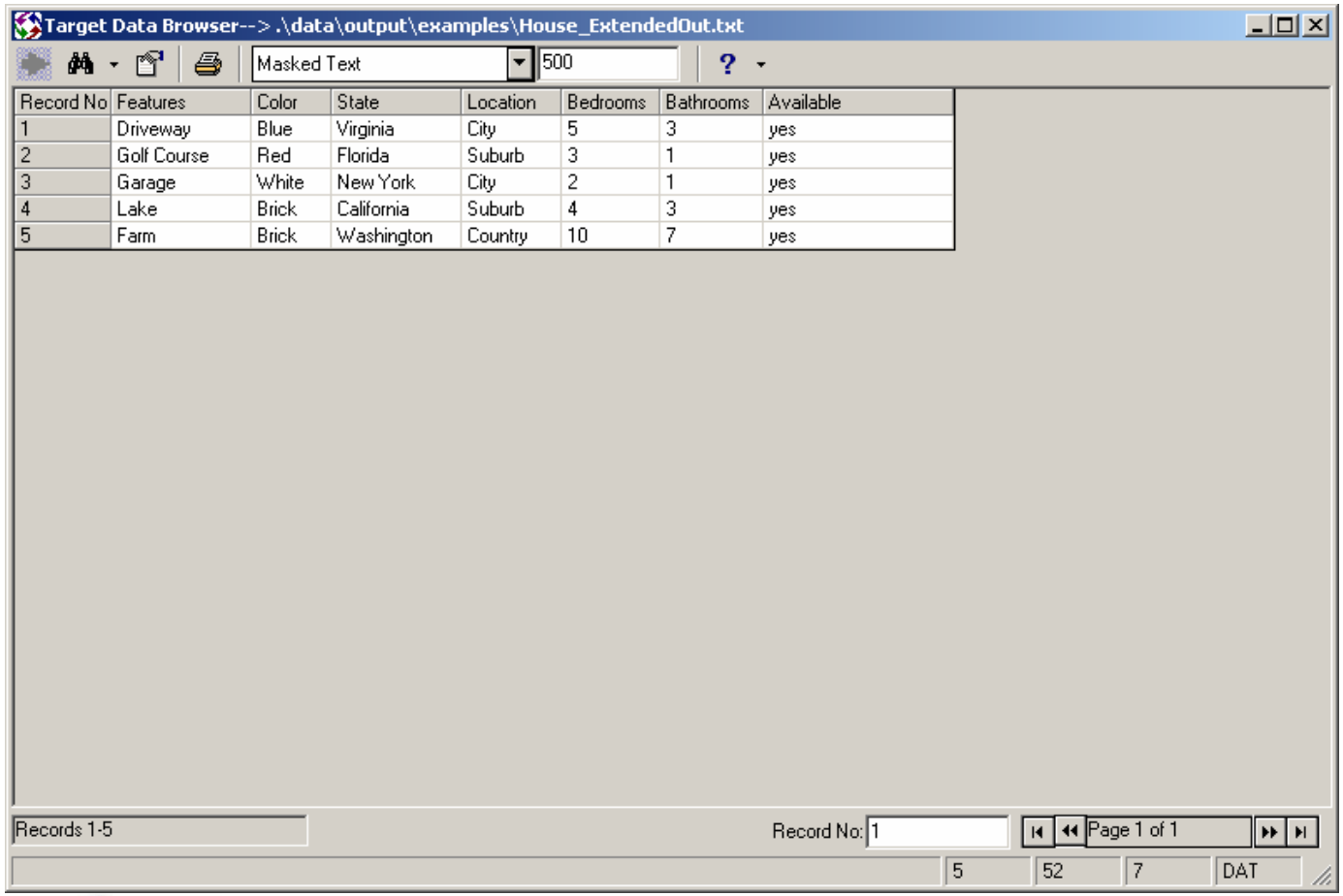


ASCII (Delimited) Properties

Property	Value
CodePage	ANSI
RecordSeparator	CR-LF
RecordFieldCount	0
FieldSeparator	tab
FieldStartDelimiter	''
FieldEndDelimiter	''
Header	False
AlternateFieldSeparator	None
StartOffset	0
AutomaticStyling	False
StyleSampleSize	5000
StripLeadingBlanks	False
StripTrailingBlanks	False
Field1IsRecTypeId	False

Exercise

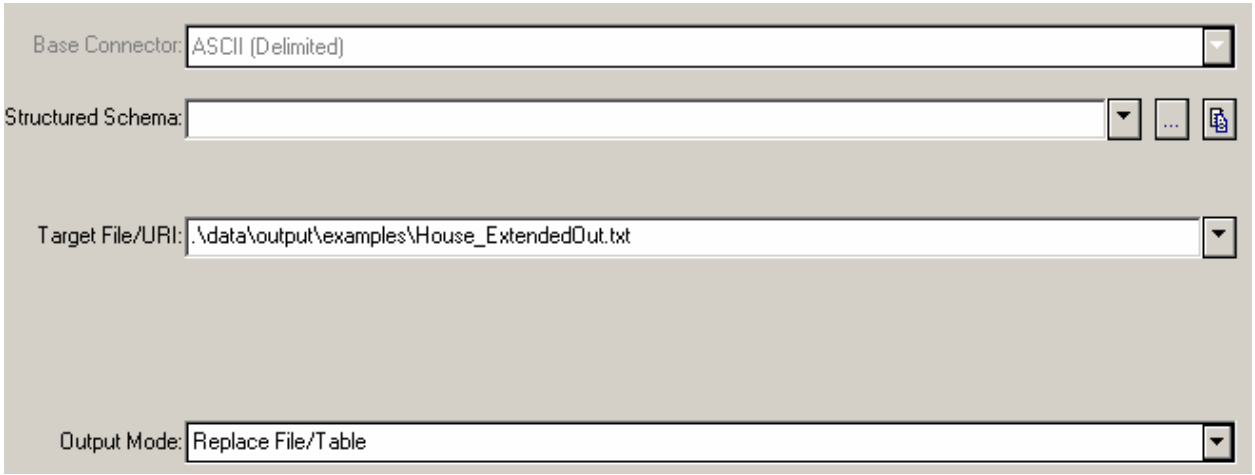
- Output File – House_ExtendedOut.txt



Record No	Features	Color	State	Location	Bedrooms	Bathrooms	Available
1	Driveway	Blue	Virginia	City	5	3	yes
2	Golf Course	Red	Florida	Suburb	3	1	yes
3	Garage	White	New York	City	2	1	yes
4	Lake	Brick	California	Suburb	4	3	yes
5	Farm	Brick	Washington	Country	10	7	yes

Exercise

- Target Properties



ASCII (Delimited) Properties

Property	Value
CodePage	ANSI
RecordSeparator	CR-LF
FieldSeparator	tab
FieldStartDelimiter	None
FieldEndDelimiter	None
Header	False
FieldDelimitStyle	all
StripLeadingBlanks	True
StripTrailingBlanks	True
TransliterationIn	
TransliterationOut	
MaxDataLen	0

Basic Conversion Steps

Conversion Overview

- ETL Concept
  - ETL stands for Extract, Transformation, and Load.
  - Separates conversion process into three steps.
  - Works well with the ADVANTAGE® applications
    - Extract using DBUTIL (2x)
    - Transformation using Data Integrator®
    - Load using JIC / SysManUtil (3x)

Basic Development Steps - DBUTIL Process

- Use delivered extract scripts to acquire data from 2.x using DBUtil for tables (Ledgers are generated by 2.x system). (DBUTIL is a common utility across all ADVANTAGE® 2.2 supported platforms)
- Other extract utilities can be used as long as files are generated mimicking the DBUTIL layout.
- Acquire the input file(s) from 2x system

Basic development process – Other interfaces

- Extract data from system
- Get layout
- Use DI to map data in 2x
- Use DI to map data in 3x
- Use the Infrastructure maps

DI Map Perform Transformation

- Prepare Mapping
- Mapping: Source
- Mapping: Target
- Event Handlers
- Miscellaneous
- Run Conversion
- Build the batch file
- Load xml through JIC / sysmanutil

Components Included in Conversion

- Map files (.map.xml)
- Code Module files (.rifl)
- Target Table/Document layouts (.dtd)
- Source files (.txt)
- SQL Statements (.sql)
- Spreadsheets (.xls)
- DOS Batch Scripts (.bat)
- JIC (.DAT)

Basic Development Steps - Load Process using JIC / SysManUtil

- Job Interaction Client (also JIC).
- Short for System Maintenance Utility (also SMU).
- Used for batch processing in ADVANTAGE® 3.x application.
  - Extracts and Loads
  - Document Processing
- Utility used on all supported platforms.
- Batch jobs can be run immediately or scheduled for later.

Prepare Mapping

Table to Table Conversion Example

One Step Process:

1. ASCII to XML

Table to Table Conversion Example

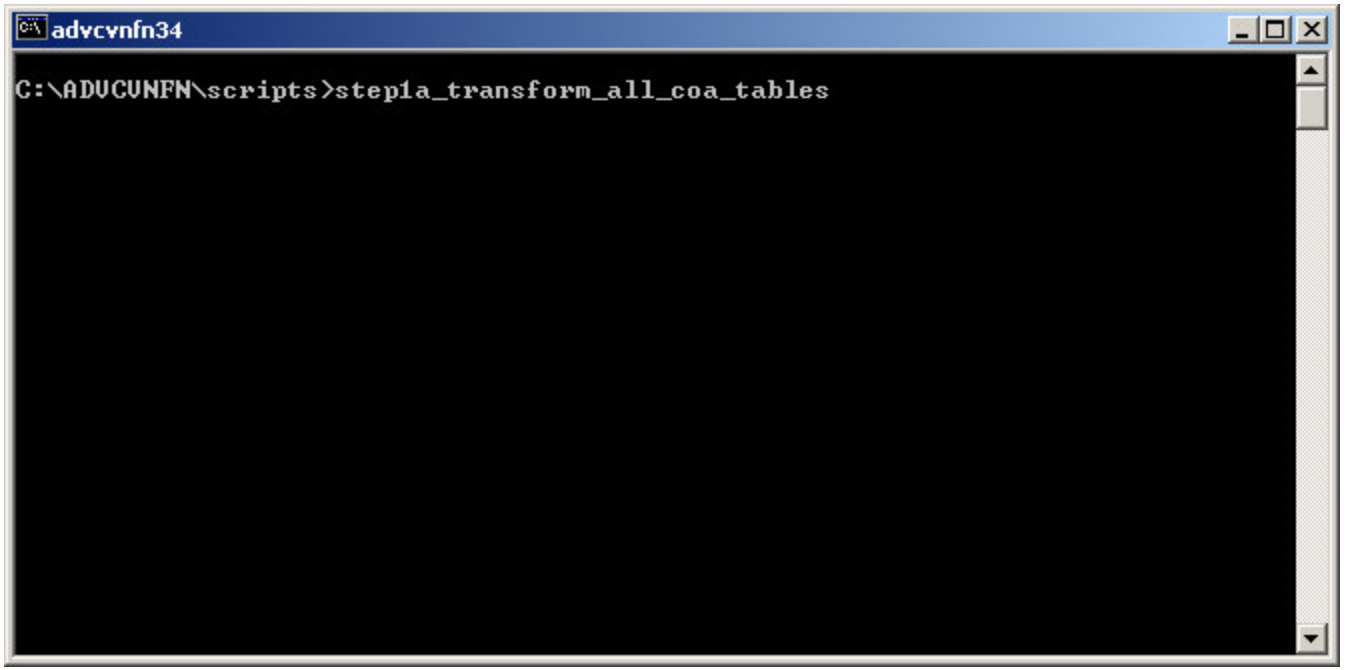


Table to Table Conversion

STEP 1: ASCII to Xml (Source Tab)

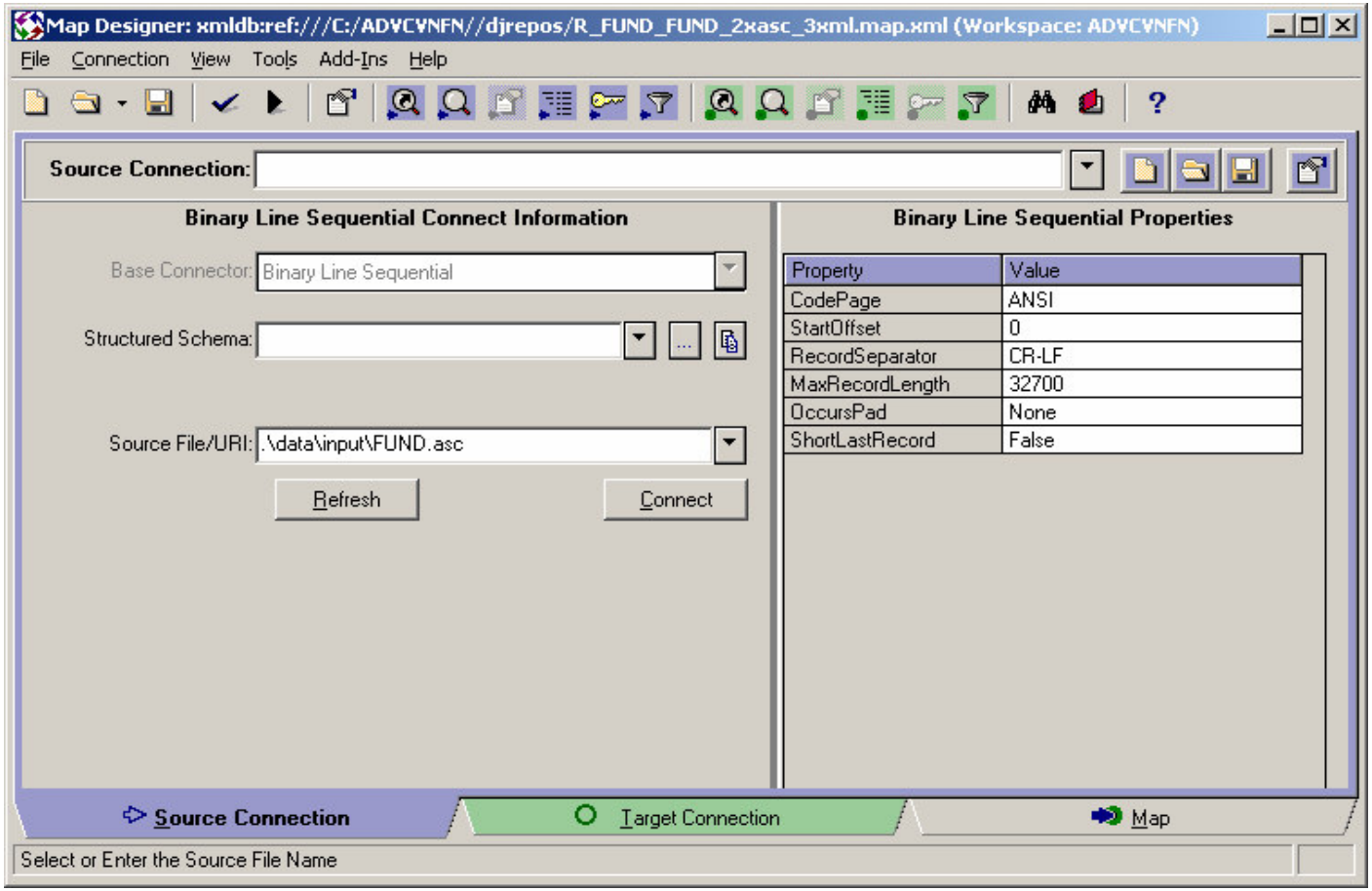


Table to Table Conversion

STEP 1: ASCII to XML (Target Tab)

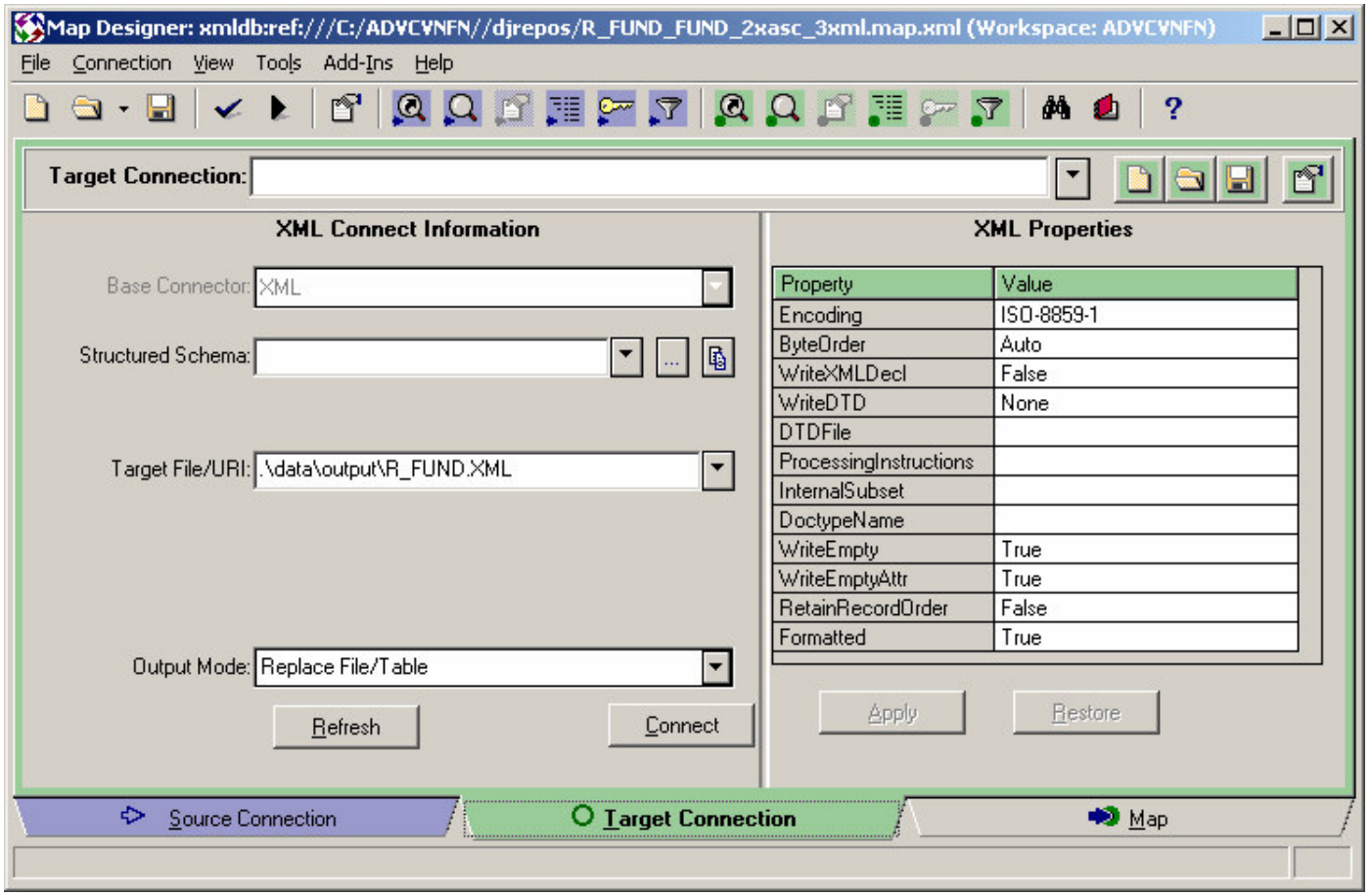


Table to Table Conversion Example Contd …

STEP 1: ASCII to XML (Map Tab)

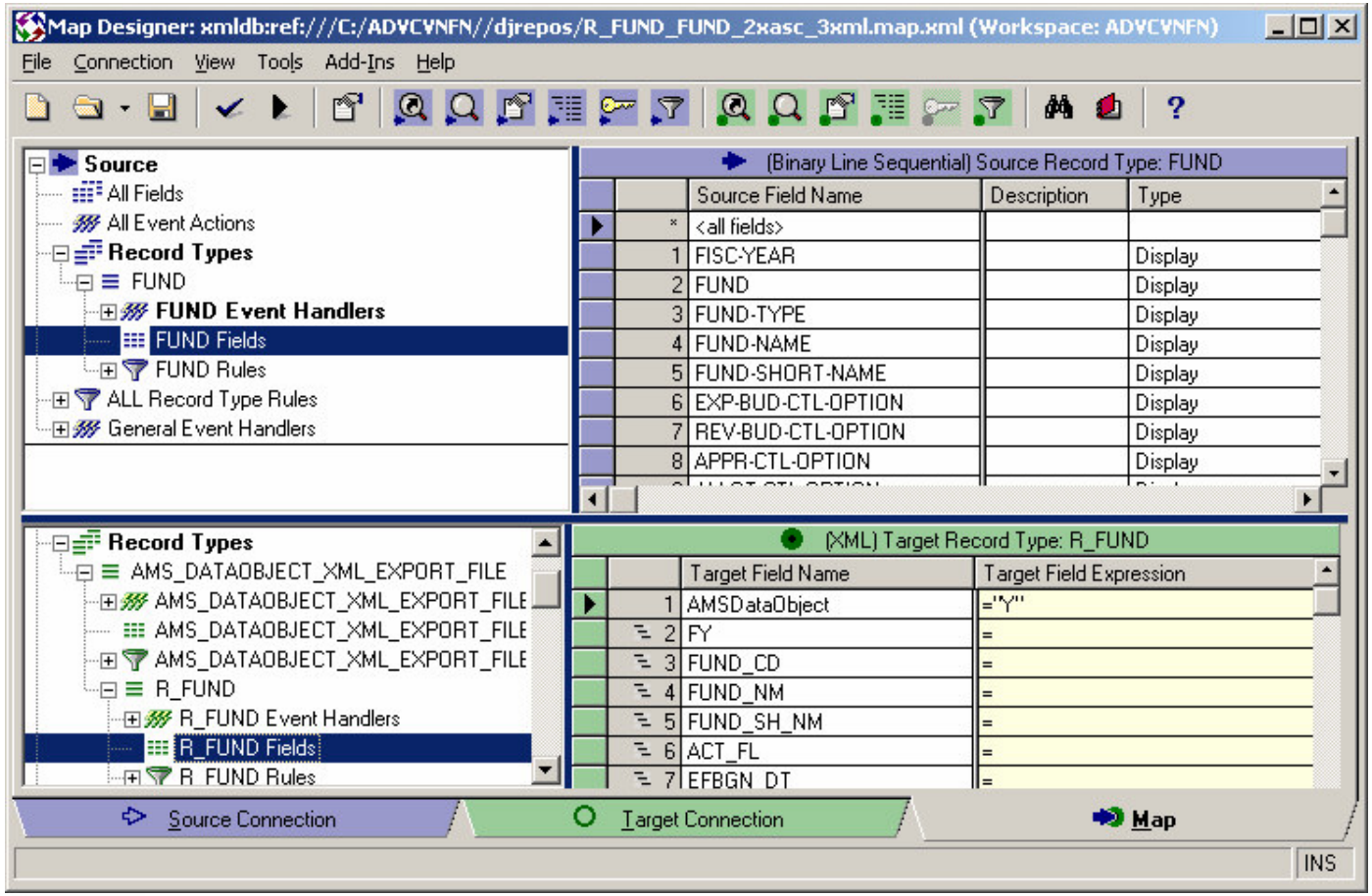


Table to Table Conversion Example Contd …

STEP 1: ASCII to XML (Events Handlers), Transformation & Map Properties

All Source Event Actions

Record Name	Event Name	No.	Action	Parameters
FUND	AfterEveryRecord	1	ClearMapPut Re	{Target},{R_FUND},{},{},{false}
FUND	AfterEveryRecord	2	ClearMapPut Re	{Target},{FY},{},{},{false}
FUND	AfterEveryRecord	3	ClearMapPut Re	{Target},{FUND_CD},{},{},{false}
FUND	AfterEveryRecord	4	ClearMapPut Re	{Target},{FUND_NM},{},{},{false}
FUND	AfterEveryRecord	5	ClearMapPut Re	{Target},{FUND_SH_NM},{},{},{fa

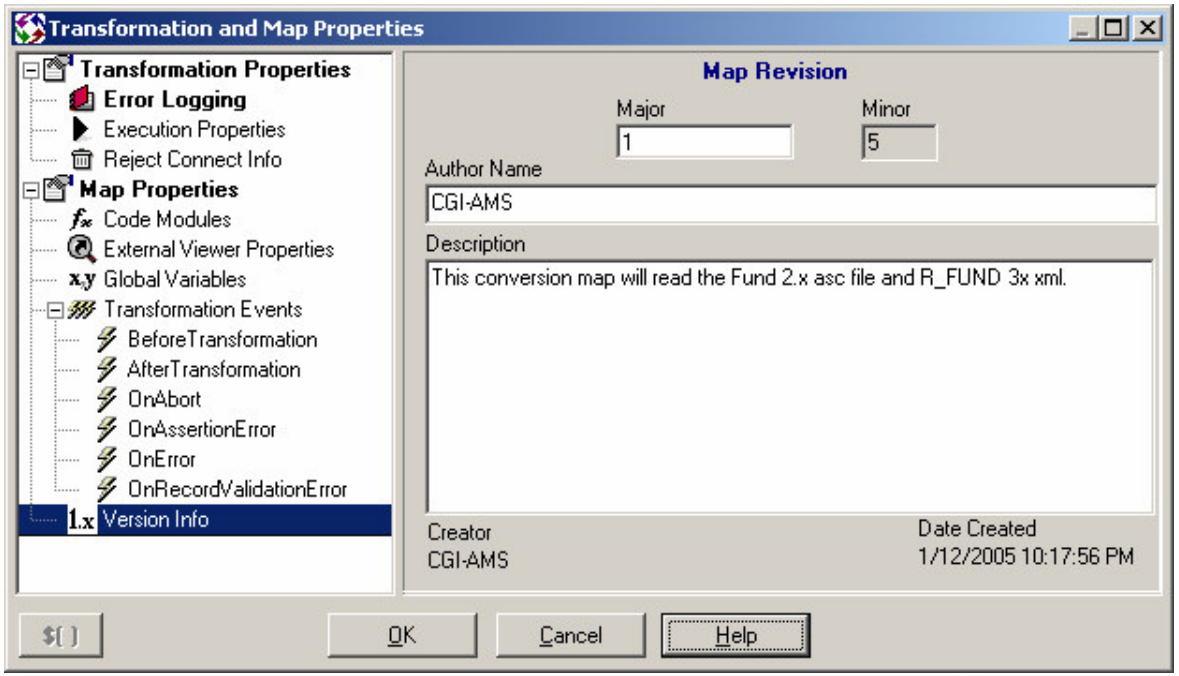
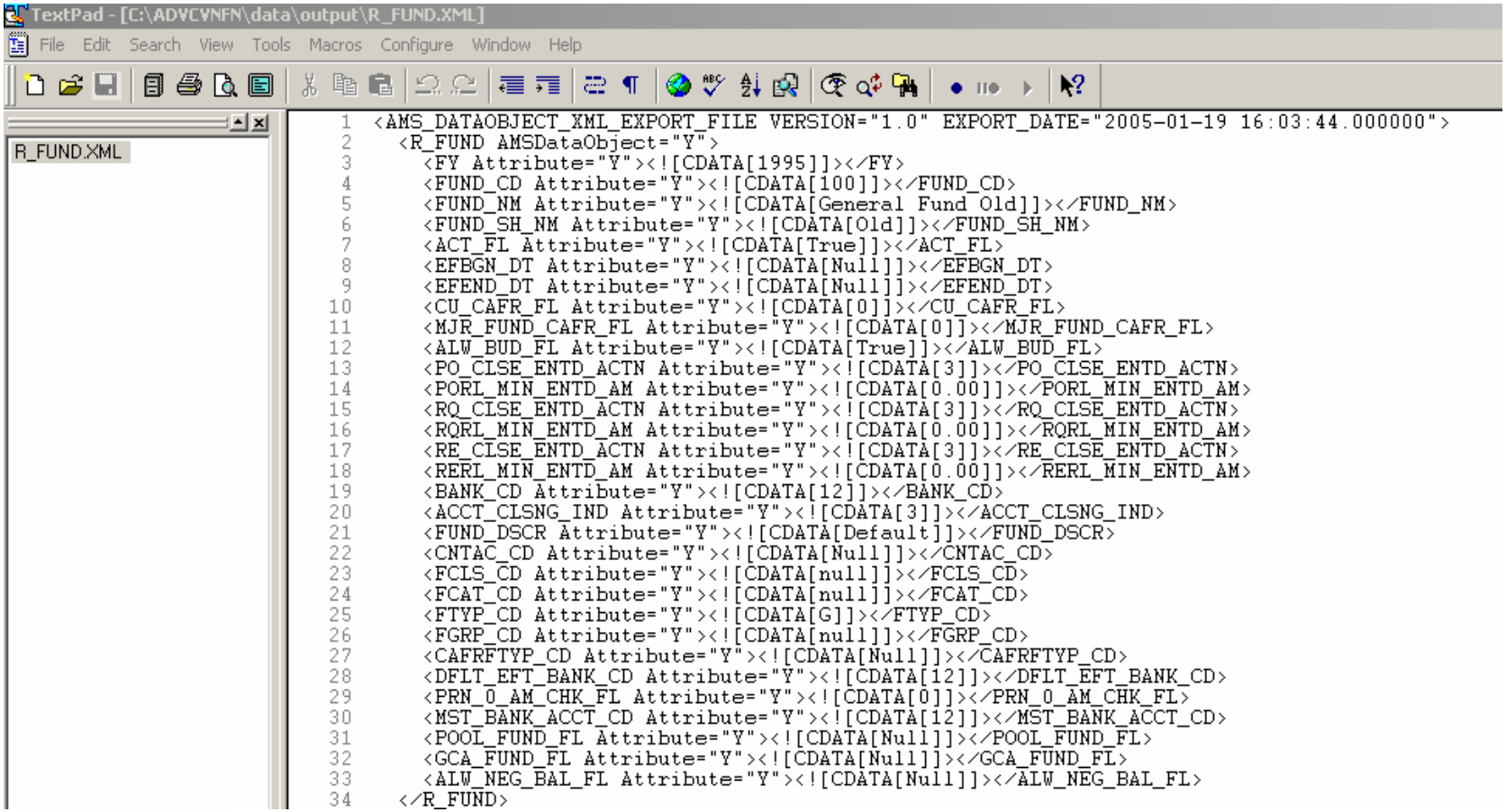


Table to Table Conversion Example Contd …

Output:



Example of JV Doc (INTERFACE)

- Run through how to take extracted data, use Data Integrator to create an XML file, load it into the application using JIC and sysmanutil
- DI Maps that are already created can be used as a starting point
- Documents, for which we do not currently have maps, use an existing map and make changes and ‘Save As’

Example of JV Document

- Two maps play a key role in the document conversion process
  - Map 1
    - 2.x format to EXTENDED-GENERAL-LEDGER
  - Map 2
    - EXTENDED-GENERAL-LEDGER to 3.x document format

Example of JV Document

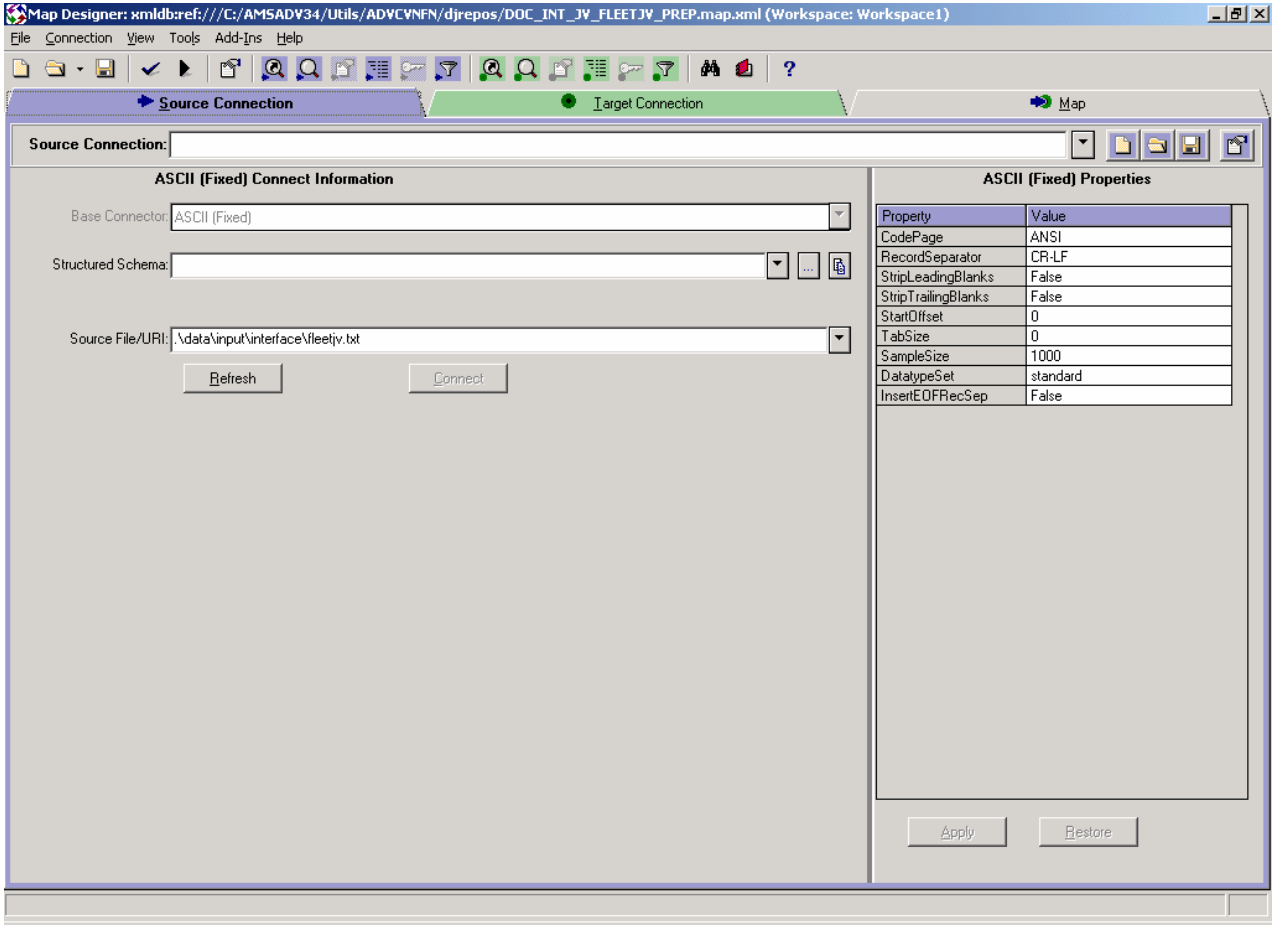
- There are supporting maps
  - DOC_INT_JV_FLEETJV_PREP
  - INF_INT_VALIDATE_LGL
  - DOC_INT_JV_JVA_MAP
  - INF_INT_RM_MAP_FAILURES
  - INF_INT_SORT
  - DOC_INT_JV_JVA_XML
  - INF_INT_DItoADV_xml_xml

JV Document Example

- Walkthrough all the Maps

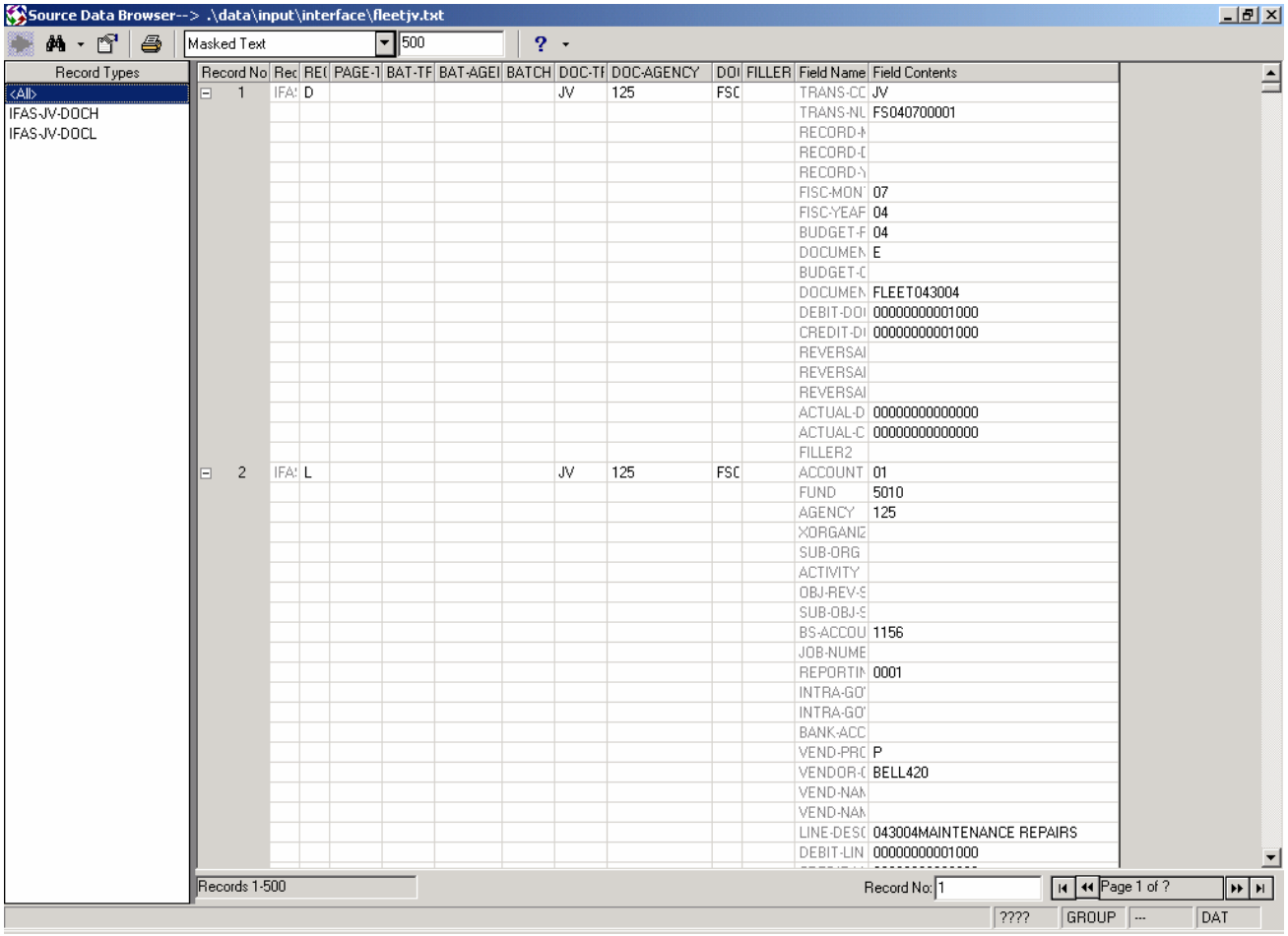
PREP Step

- DOC_INT_JV_FLEETJV_PREP



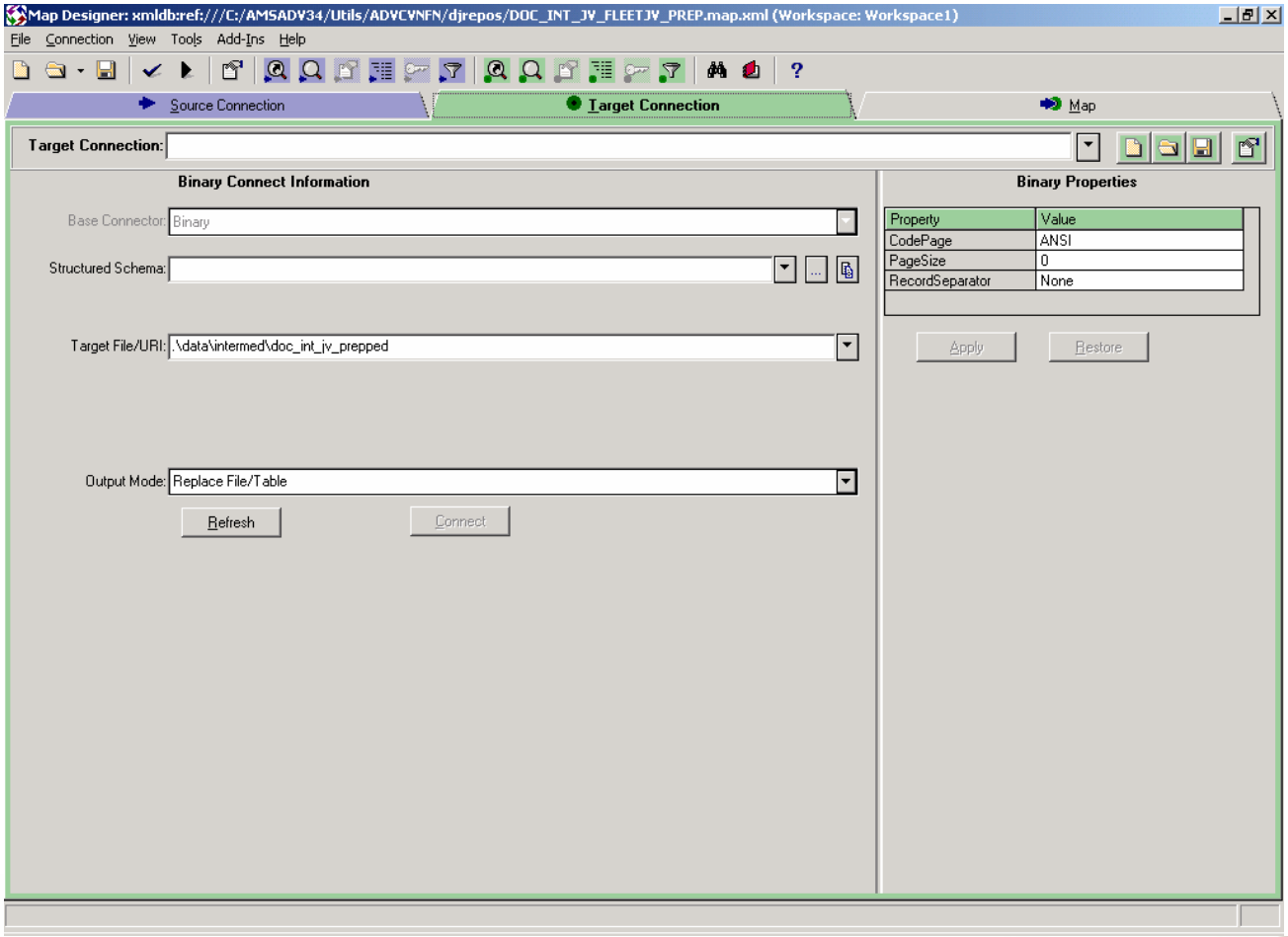
View Source Data

- Look at the source data by opening the Source Data Browser



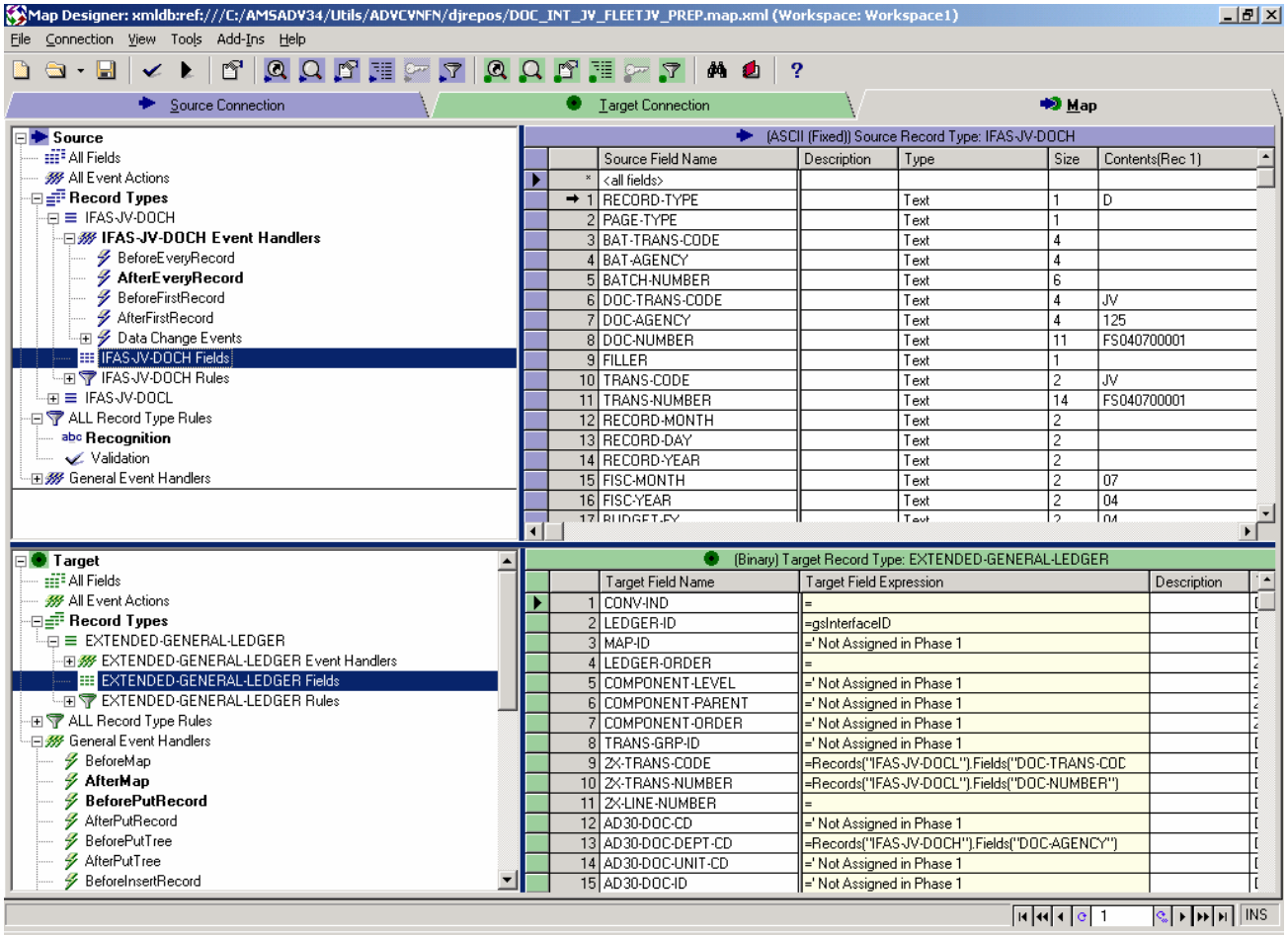
Target Tab

- Navigate to the Target Tab

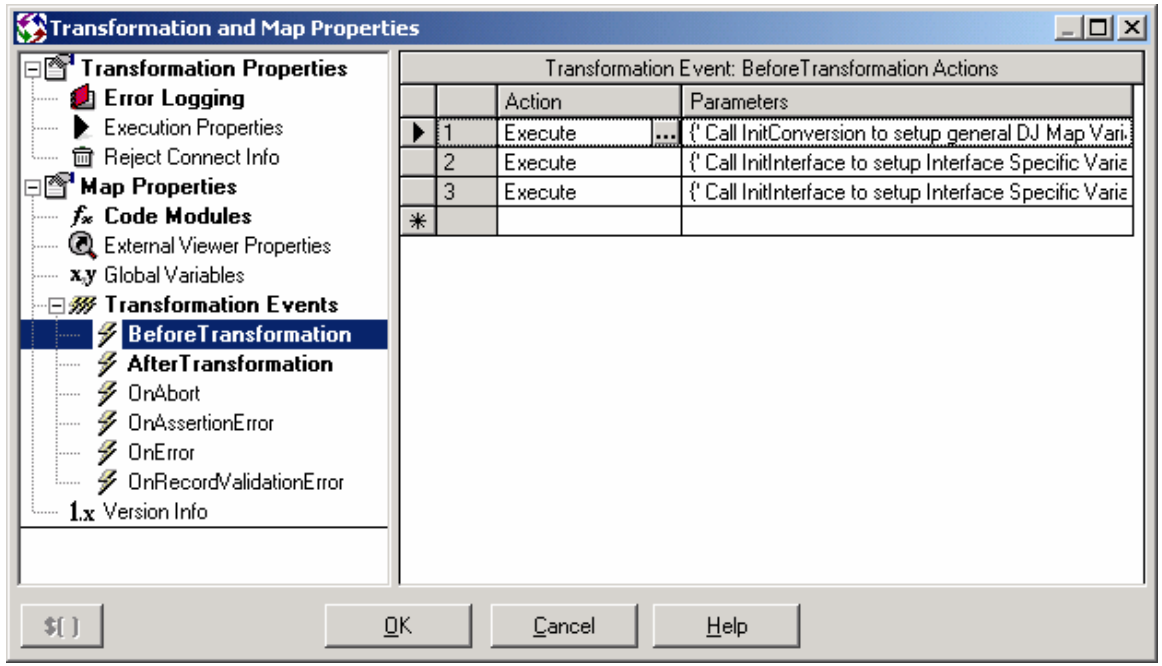


Map Tab

- Navigate to the Map Tab



View Transformation and Map Properties



View Source Rules/Actions…

- Record Type Rules

Record Discriminator: Records("IFAS-JV-DOCH").Fields("RECORD-TYPE" — Generate Rules

(ASCII (Fixed)) Source Record Recognition Rules, Discriminator--> Data Type: Text, Offset: 1, Length: 1

	Discr	Operator	Value	High Value	Case	Record Type	Rule Name	Description
1	Discri	=	D		☑	IFAS-JV-DOCH	RecordRule1	
2	Discri	=	L		☑	IFAS-JV-DOCL	RecordRule2	
*					☐			

- All Source Event Actions

All Source Event Actions

Record Name	Event Name	No.	Action	Parameters
IFAS-JV-DOCH	AfterEveryRecord	1	Clear	{Target},{EXTENDED-GENERAL
IFAS-JV-DOCH	AfterEveryRecord	2	Execute	{giNumDocs = giNumDocs + 1|' Ir
IFAS-JV-DOCL	AfterEveryRecord	1	MapPut Record	{Target},{EXTENDED-GENERAL
*				

View Target Actions … Contd …

- All Target Event Actions

All Target Event Actions

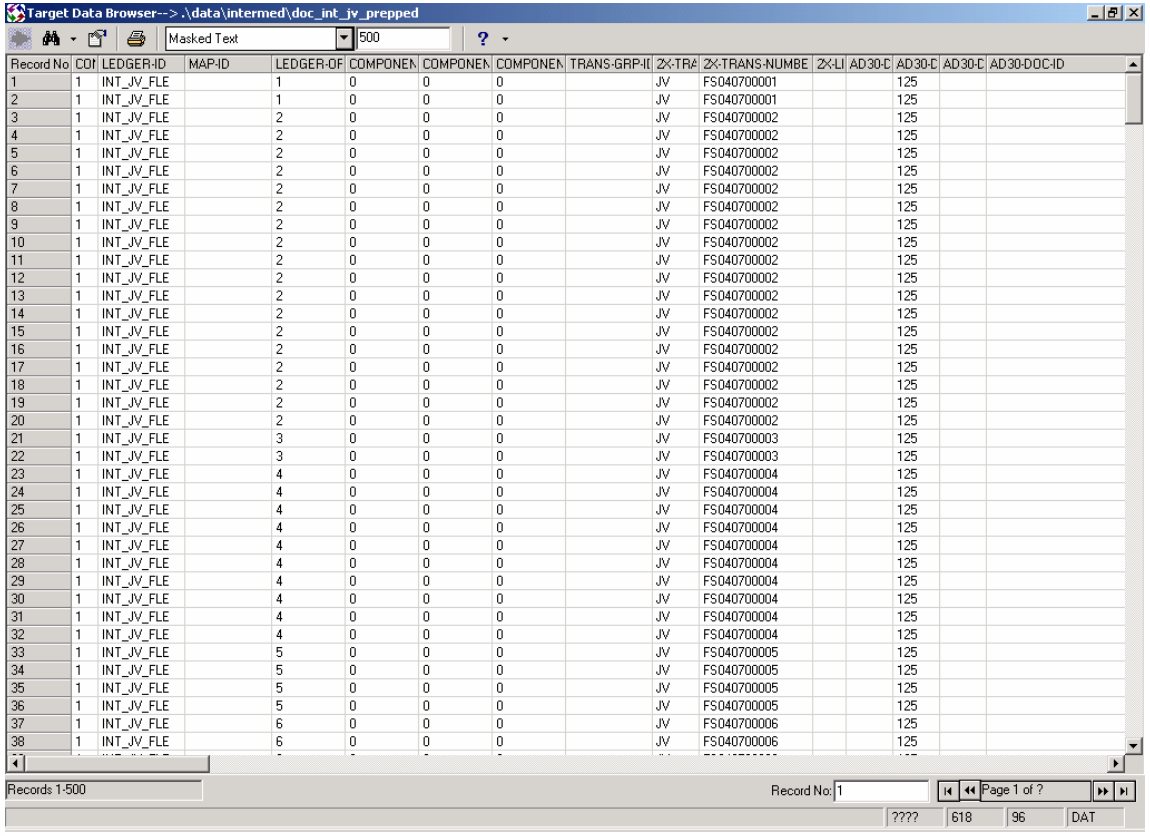
	Record Name	Event Name	No.	Action	Parameters
▶		AfterMap	1	Execute	{If Targets(0).Records("EXTENC
		BeforePutRecord	1	Execute	{Targets(0).Records("EXTENDEC
*					

- gbCommitInterfaceTrans

```
giNumDocs = giNumDocs + 1
' Initialize Keep Status for each new document to True (Anything later can set this flag to false
gbCommitInterfaceTrans = 1
```

View Target Data

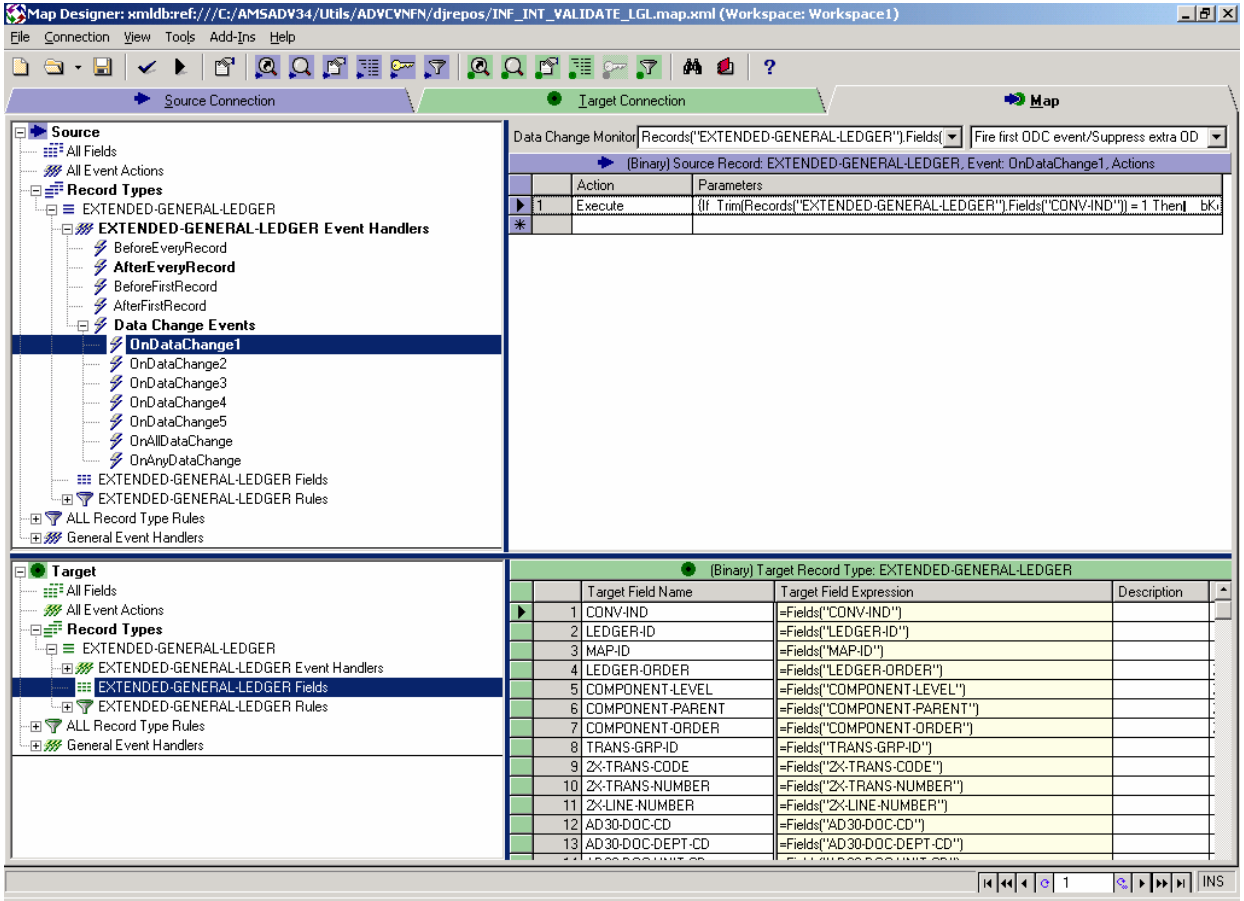
- Look at the target data by opening the Target Data Browser



Record No	COI	LEDGER-ID	MAP-ID	LEDGER-OF	COMPONEN	COMPONEN	COMPONEN	TRANS-GRP-II	ZX-TRA	ZX-TRANS-NUMBE	ZX-LI	AD30-D	AD30-D	AD30-D	AD30-DOC-ID
1	1	INT_JV_FLE		1	0	0	0		JV	FS040700001			125		
2	1	INT_JV_FLE		1	0	0	0		JV	FS040700001			125		
3	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
4	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
5	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
6	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
7	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
8	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
9	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
10	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
11	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
12	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
13	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
14	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
15	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
16	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
17	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
18	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
19	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
20	1	INT_JV_FLE		2	0	0	0		JV	FS040700002			125		
21	1	INT_JV_FLE		3	0	0	0		JV	FS040700003			125		
22	1	INT_JV_FLE		3	0	0	0		JV	FS040700003			125		
23	1	INT_JV_FLE		4	0	0	0		JV	FS040700004			125		
24	1	INT_JV_FLE		4	0	0	0		JV	FS040700004			125		
25	1	INT_JV_FLE		4	0	0	0		JV	FS040700004			125		
26	1	INT_JV_FLE		4	0	0	0		JV	FS040700004			125		
27	1	INT_JV_FLE		4	0	0	0		JV	FS040700004			125		
28	1	INT_JV_FLE		4	0	0	0		JV	FS040700004			125		
29	1	INT_JV_FLE		4	0	0	0		JV	FS040700004			125		
30	1	INT_JV_FLE		4	0	0	0		JV	FS040700004			125		
31	1	INT_JV_FLE		4	0	0	0		JV	FS040700004			125		
32	1	INT_JV_FLE		4	0	0	0		JV	FS040700004			125		
33	1	INT_JV_FLE		5	0	0	0		JV	FS040700005			125		
34	1	INT_JV_FLE		5	0	0	0		JV	FS040700005			125		
35	1	INT_JV_FLE		5	0	0	0		JV	FS040700005			125		
36	1	INT_JV_FLE		5	0	0	0		JV	FS040700005			125		
37	1	INT_JV_FLE		6	0	0	0		JV	FS040700006			125		
38	1	INT_JV_FLE		6	0	0	0		JV	FS040700006			125		

VALIDATE Step

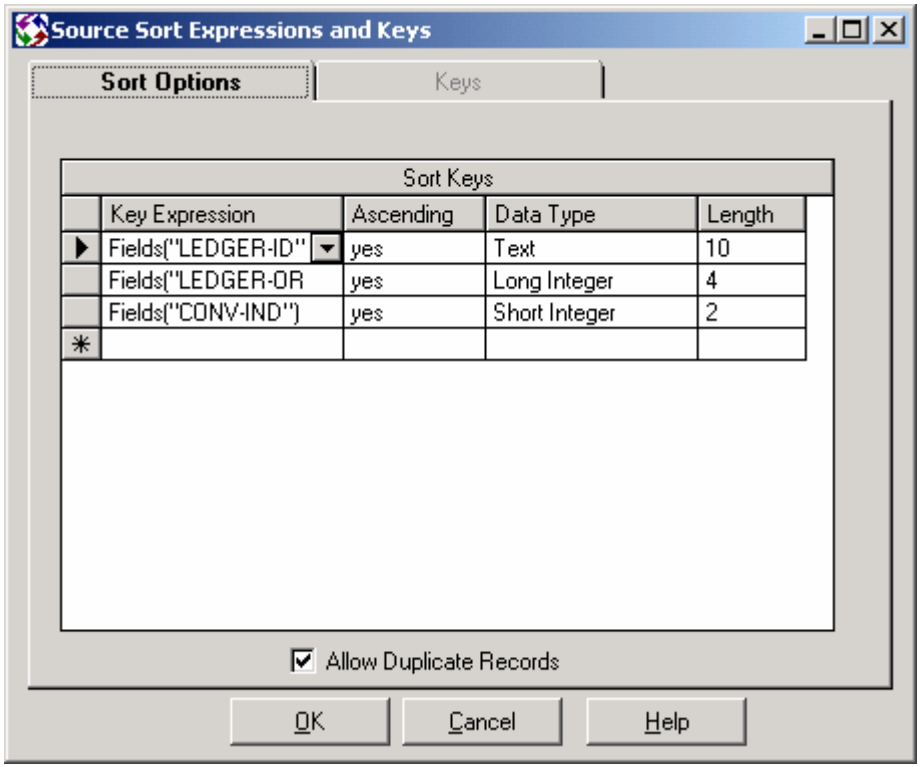
- INF_INT_VALIDATE_LGL (NO NEED TO CHANGE)



VALIDATE Step Contd …

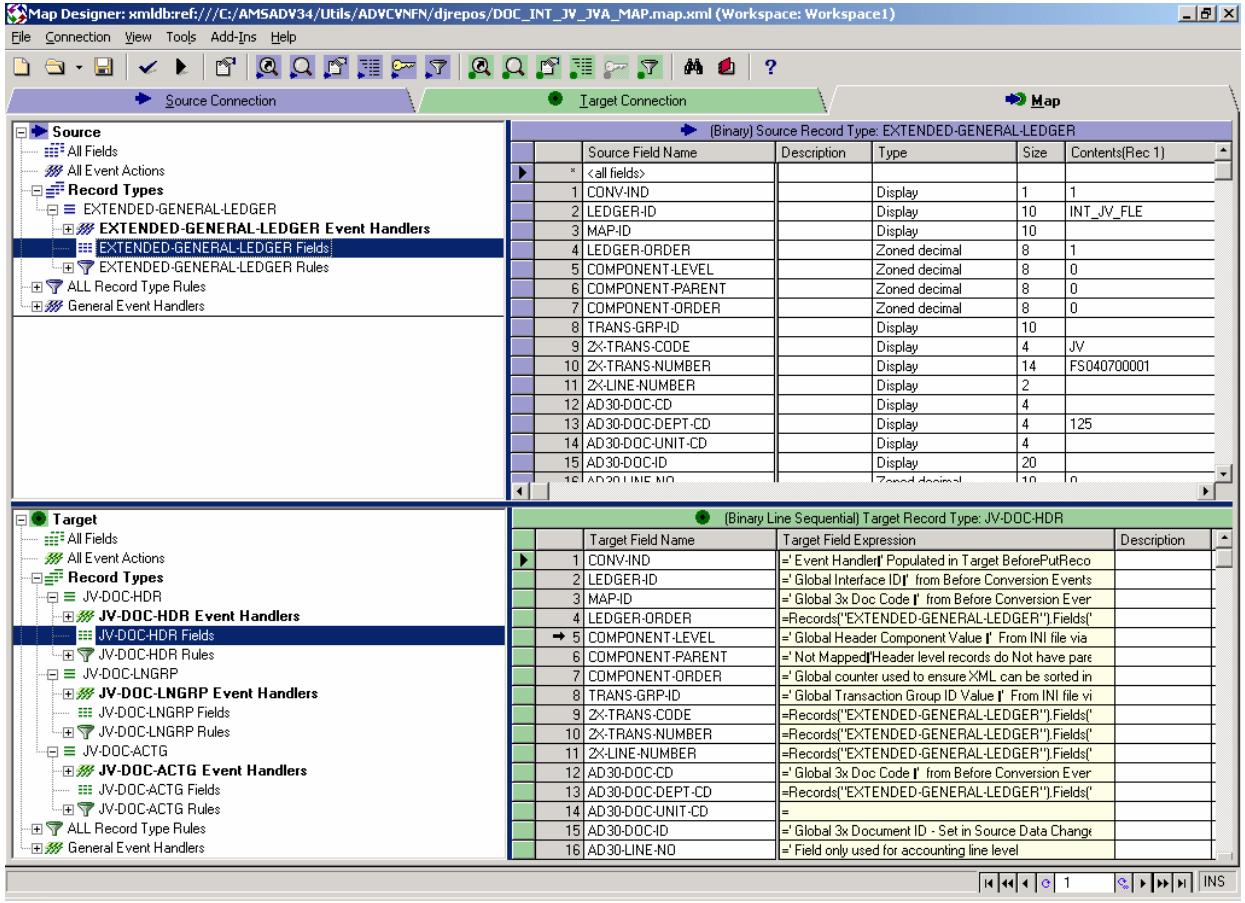
- INF_INT_VALIDATE_LGL (NO NEED TO CHANGE)

```
If  Trim(Records("EXTENDED-GENERAL-LEDGER").Fields("CONV-IND")) = 1 Then
    bKeepDocument = True
Else
    bKeepDocument = False
End If
```



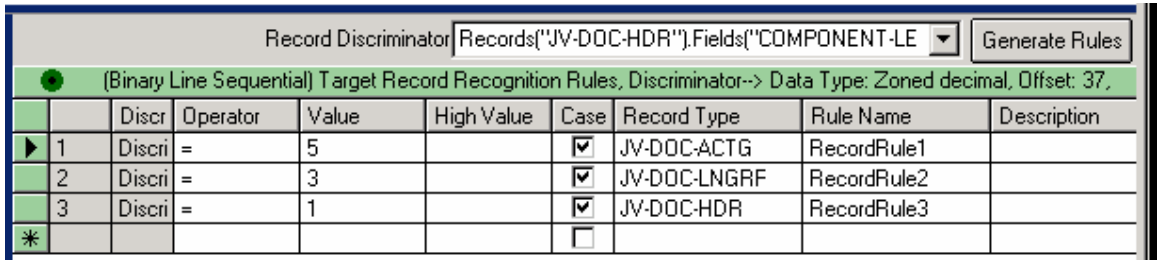
MAP Step

- DOC_INT_JV_JVA_MAP



View Target Rules …

- Record Type Rules



		Discr	Operator	Value	High Value	Case	Record Type	Rule Name	Description
▶	1	Discri	=	5		☑	JV-DOC-ACTG	RecordRule1	
	2	Discri	=	3		☑	JV-DOC-LNGRF	RecordRule2	
	3	Discri	=	1		☑	JV-DOC-HDR	RecordRule3	
*						☐			

Source and Target Event Actions

- Source Event Actions

	Record Name	Event Name	No.	Action	Parameters
▶		AfterEveryRecord	1	Execute	{' Increament Lowest Level Child (
		AfterEveryRecord	2	ClearMapPut Re	{Target},{JV-DOC-ACTG},{},{},{}
	EXTENDED-GENE	OnDataChange1	1	Put Record	{Target},{JV-DOC-LNGRP},{},{},{}
	EXTENDED-GENE	OnDataChange1	2	Execute	{' Increment the LNGRP here in th
	EXTENDED-GENE	OnDataChange2	1	Execute	{' Increment Global Variable - Acc
	EXTENDED-GENE	OnDataChange2	2	ClearMap	{Target},{JV-DOC-LNGRP},{},{}
	EXTENDED-GENE	OnDataChange3	1	Put Record	{Target},{JV-DOC-HDR},{},{},{}
	EXTENDED-GENE	OnDataChange3	2	Execute	{' Reset Dcoument Child Compone
	EXTENDED-GENE	OnDataChange4	1	Execute	{' Increment Global Variable - Num
	EXTENDED-GENE	OnDataChange4	2	ClearMap	{Target},{JV-DOC-HDR},{},{}
*					

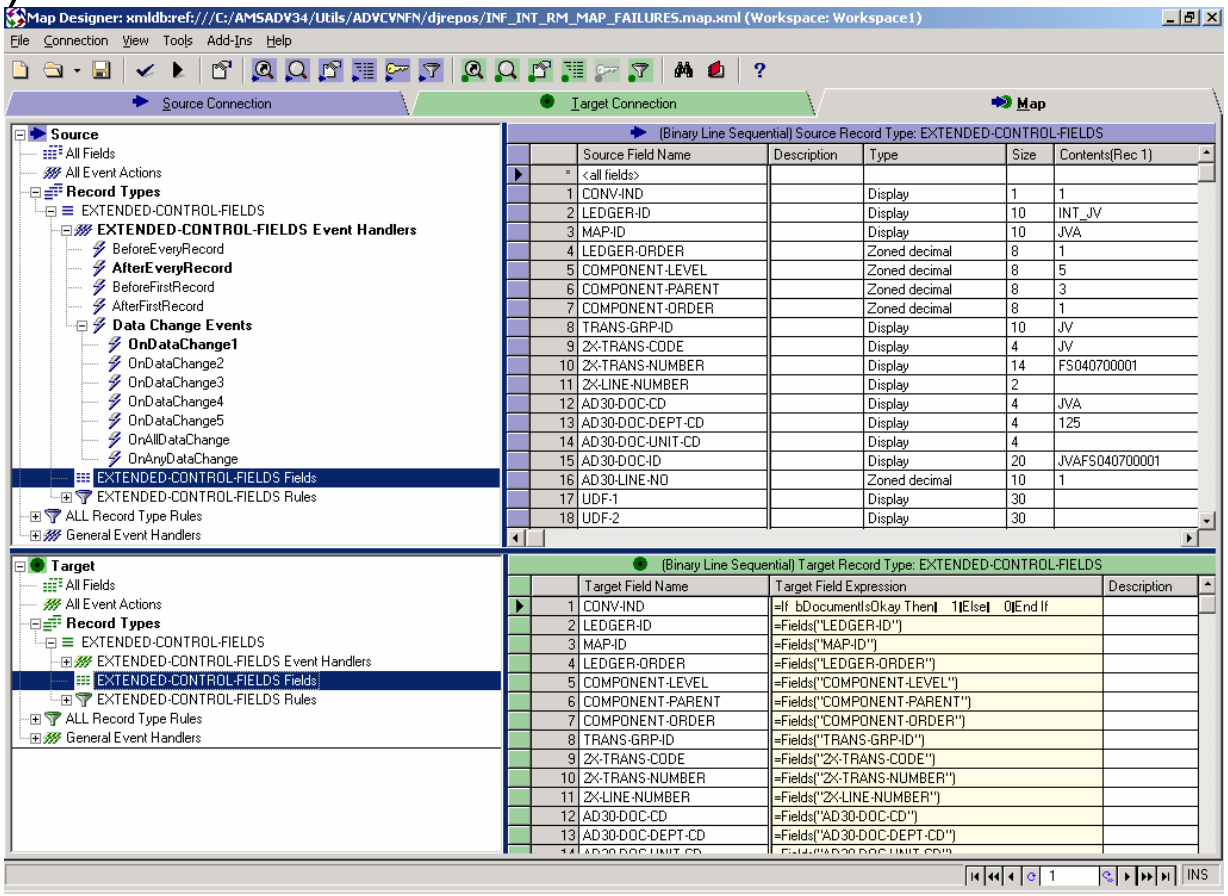
- Target Event Actions

All Target Event Actions

	Record Name	Event Name	No.	Action	Parameters
▶	JV-DOC-ACTG	BeforePutRecord	1	Execute	{' Infrastructure Counter Field for X
	JV-DOC-ACTG	BeforePutRecord	2	Execute	{' Record the document level rejec
	JV-DOC-HDR	BeforePutRecord	1	Execute	{' Infrastructure Counter Field for X
	JV-DOC-HDR	BeforePutRecord	2	Execute	{' Record the document level rejec
	JV-DOC-LNGRP	BeforePutRecord	1	Execute	{' Infrastructure Counter Field for X
	JV-DOC-LNGRP	BeforePutRecord	2	Execute	{' Record the document level rejec
*					

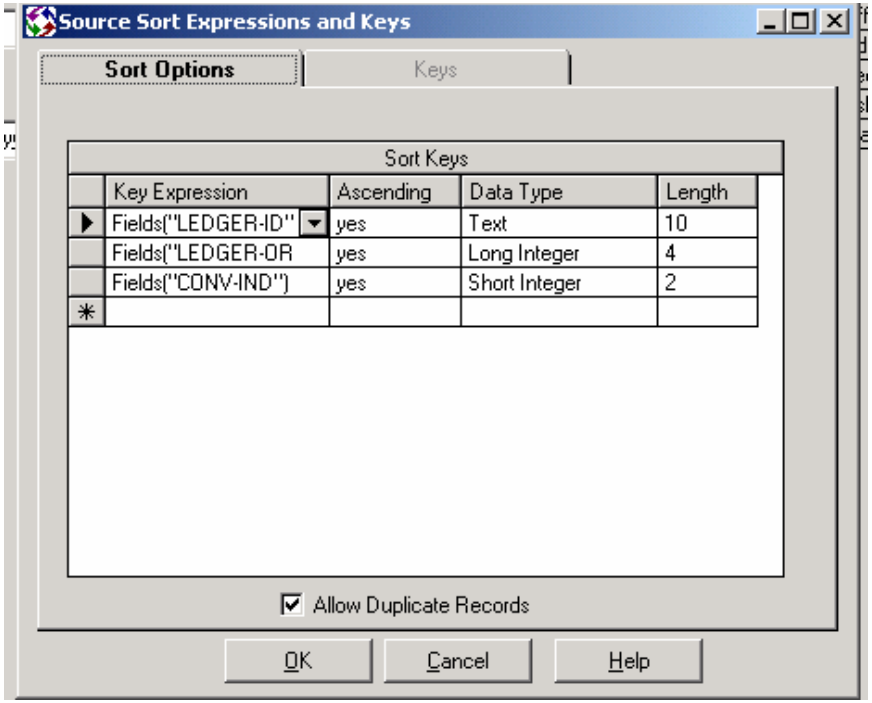
Map Failures Step

- INF_INT_RM_MAP_FAILURES (NO NEED TO CHANGE)



Map Failures Step Contd …

- INF_INT_RM_MAP_FAILURES (NO NEED TO CHANGE)

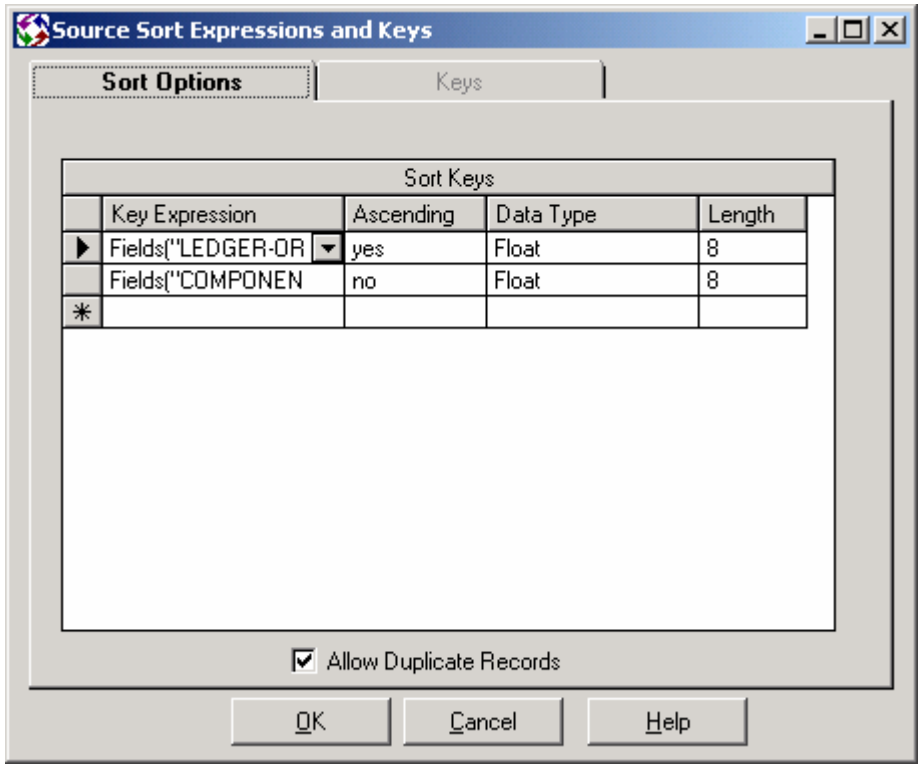


- bDocumentIsOkay

```
If  Trim(Records("EXTENDED-CONTROL-FIELDS").Fields("CONV-IND")) = "1" Then
    bDocumentIsOkay = True
Else
    bDocumentIsOkay = False
End If
```

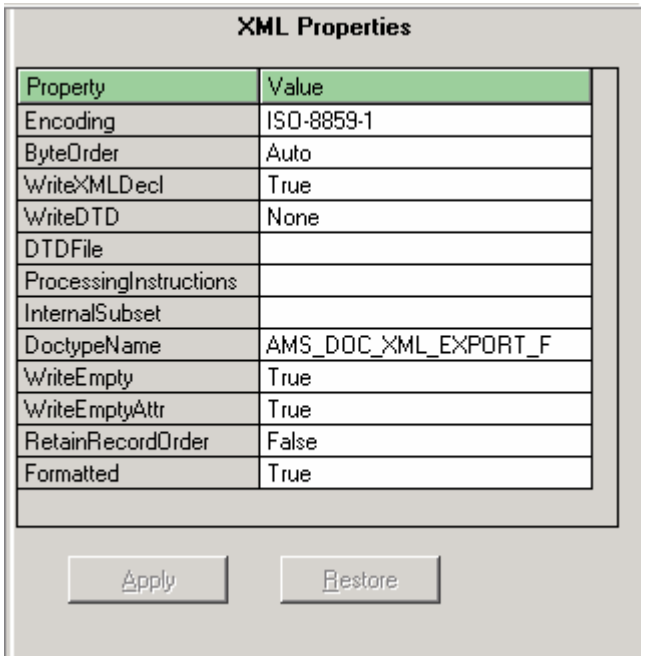
Sort Step

- INF_INT_SORT (NO NEED TO CHANGE)



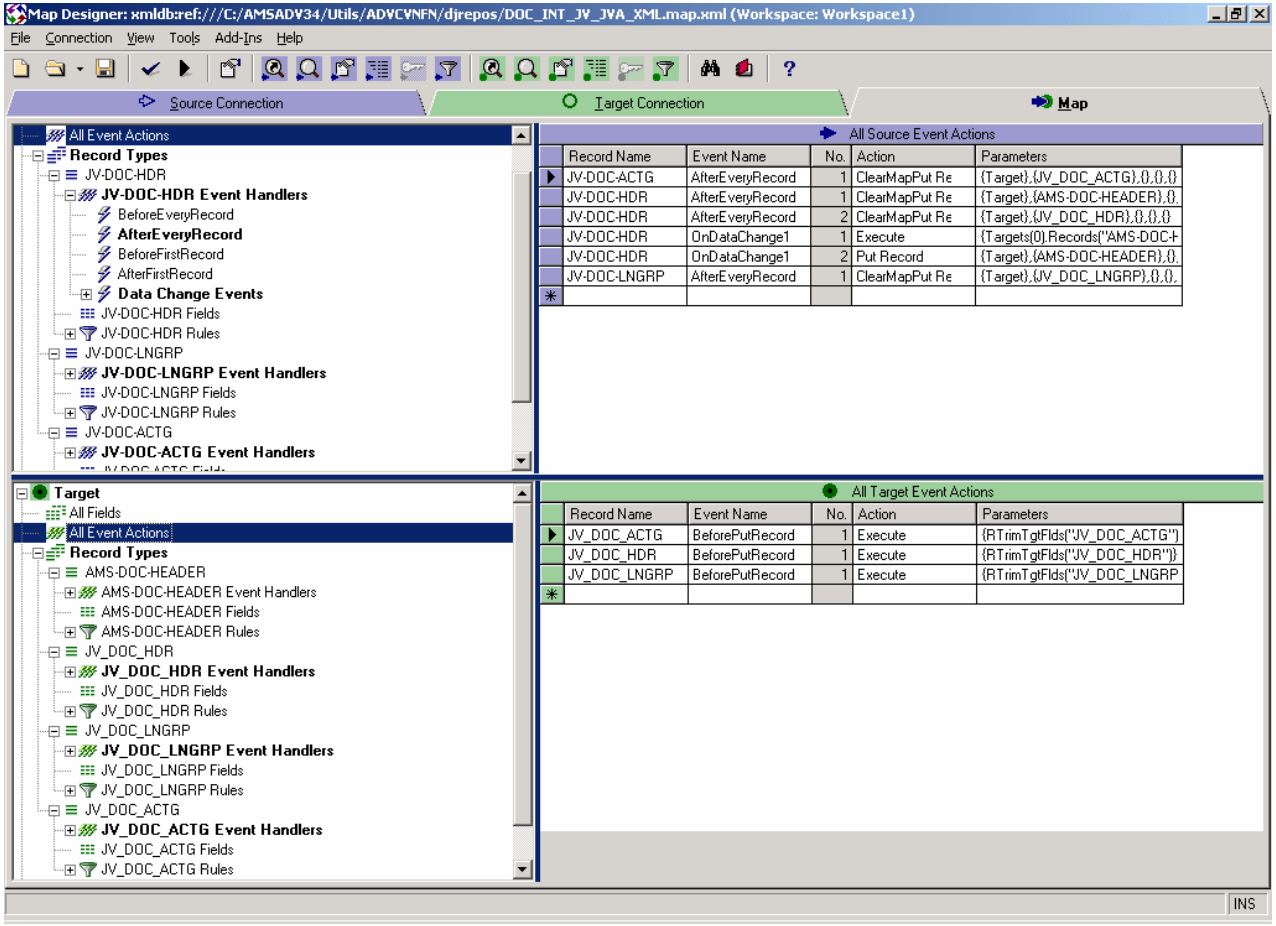
XML Step

- DOC_INT_JV_JVA_XML



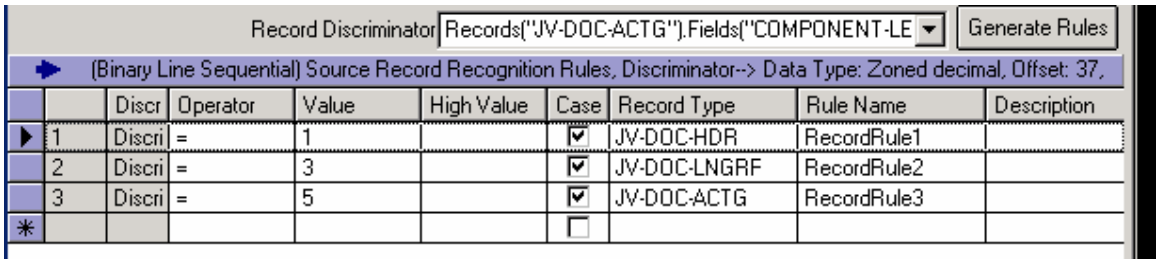
XML Step Contd …

- DOC_INT_JV_JVA_XML



XML Step Contd …

- Record Type Rules

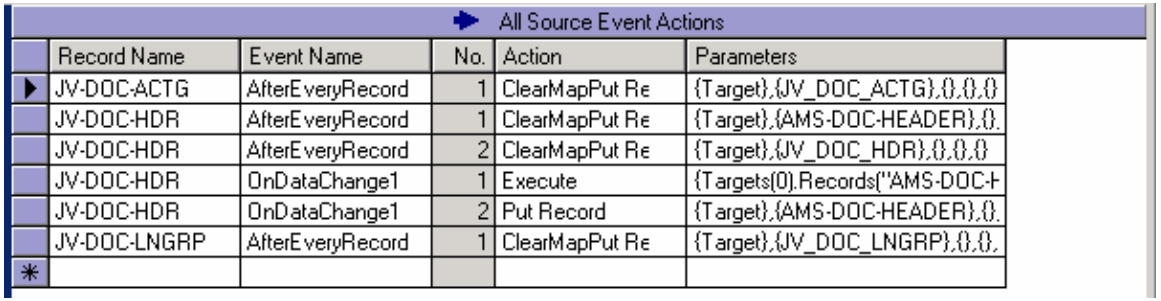


Record Discriminator: Records("JV-DOC-ACTG").Fields("COMPONENT-LE — Generate Rules

(Binary Line Sequential) Source Record Recognition Rules, Discriminator--> Data Type: Zoned decimal, Offset: 37,

	Discr	Operator	Value	High Value	Case	Record Type	Rule Name	Description
1	Discri	=	1		☑	JV-DOC-HDR	RecordRule1	
2	Discri	=	3		☑	JV-DOC-LNGRF	RecordRule2	
3	Discri	=	5		☑	JV-DOC-ACTG	RecordRule3	
*					☐			

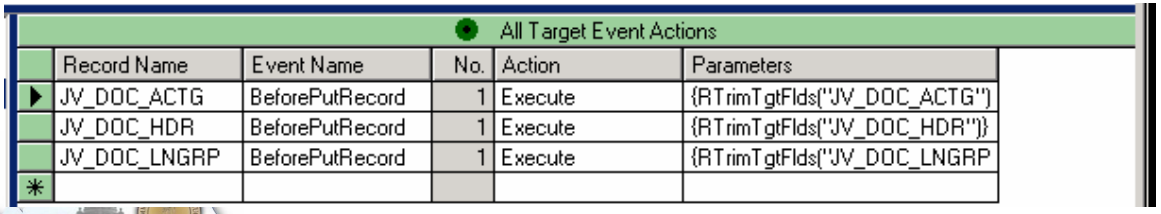
- Source Event Actions



All Source Event Actions

Record Name	Event Name	No.	Action	Parameters
JV-DOC-ACTG	AfterEveryRecord	1	ClearMapPut Re	{Target},{JV_DOC_ACTG},{},{},{}
JV-DOC-HDR	AfterEveryRecord	1	ClearMapPut Re	{Target},{AMS-DOC-HEADER},{},
JV-DOC-HDR	AfterEveryRecord	2	ClearMapPut Re	{Target},{JV_DOC_HDR},{},{},{}
JV-DOC-HDR	OnDataChange1	1	Execute	{Targets(0).Records("AMS-DOC-H
JV-DOC-HDR	OnDataChange1	2	Put Record	{Target},{AMS-DOC-HEADER},{},
JV-DOC-LNGRP	AfterEveryRecord	1	ClearMapPut Re	{Target},{JV_DOC_LNGRP},{},{},
*				

- Target Event Actions

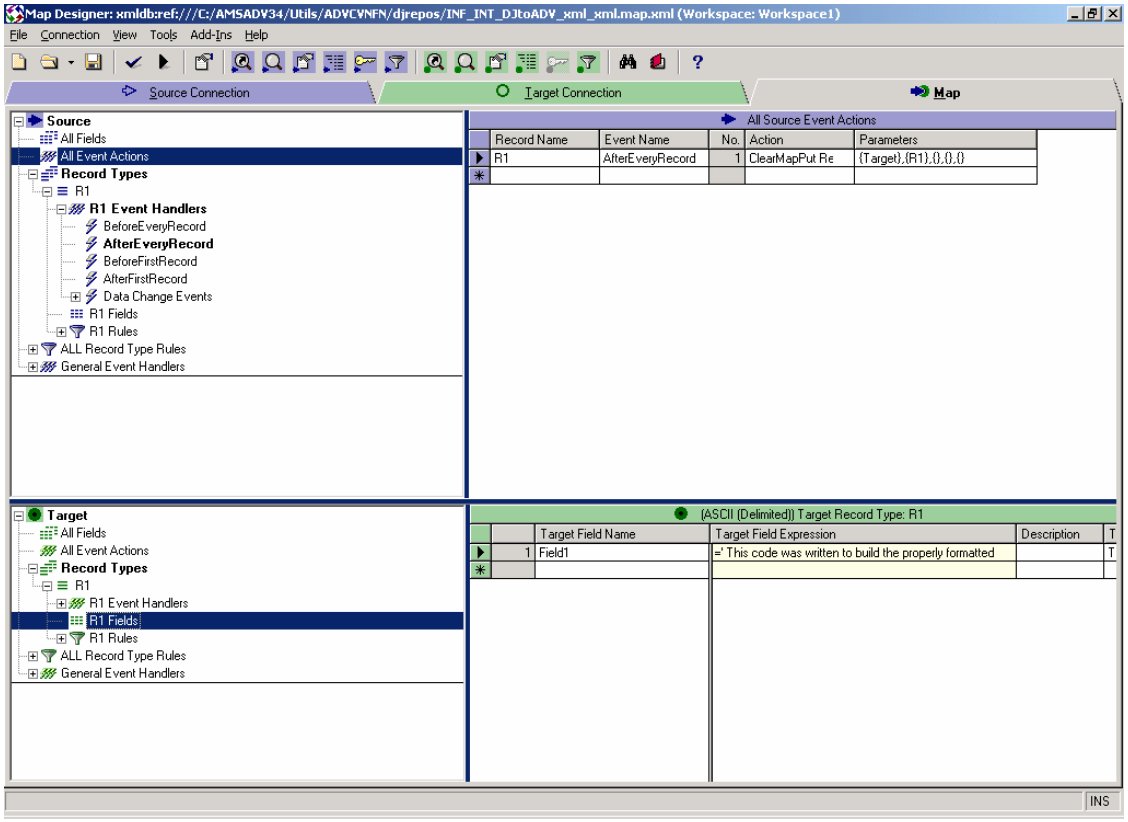


All Target Event Actions

Record Name	Event Name	No.	Action	Parameters
JV_DOC_ACTG	BeforePutRecord	1	Execute	{RTrimTgtFlds("JV_DOC_ACTG")
JV_DOC_HDR	BeforePutRecord	1	Execute	{RTrimTgtFlds("JV_DOC_HDR")}
JV_DOC_LNGRP	BeforePutRecord	1	Execute	{RTrimTgtFlds("JV_DOC_LNGRP
*				

INFRASTRUCTURE XML Step

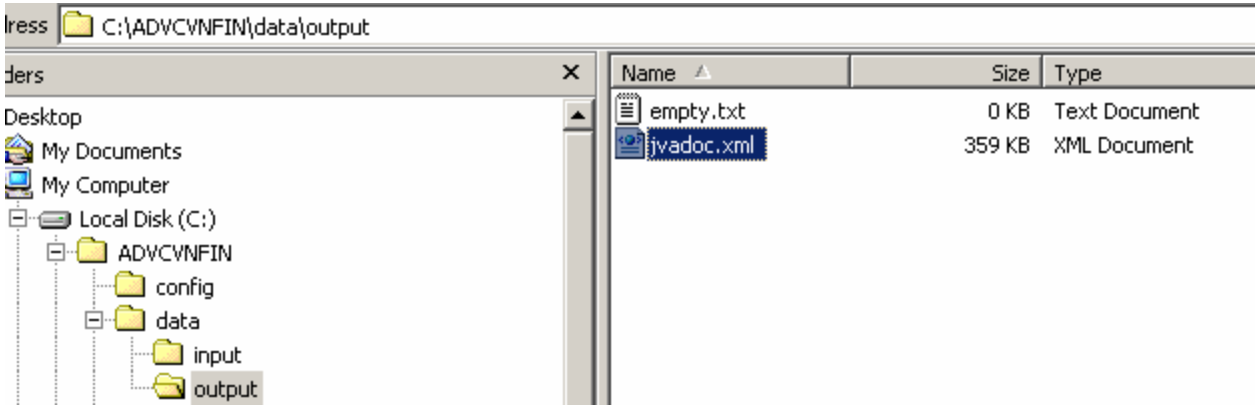
- INF_INT_DItoADV_xml_xml (NO NEED TO CHANGE)




```
sDefaultDateTime = now()
sDataObjectName = "##############"
sEndFieldsMarker = "##############"
sAttributeString = " Attribute=""Y""><![CDATA["
```

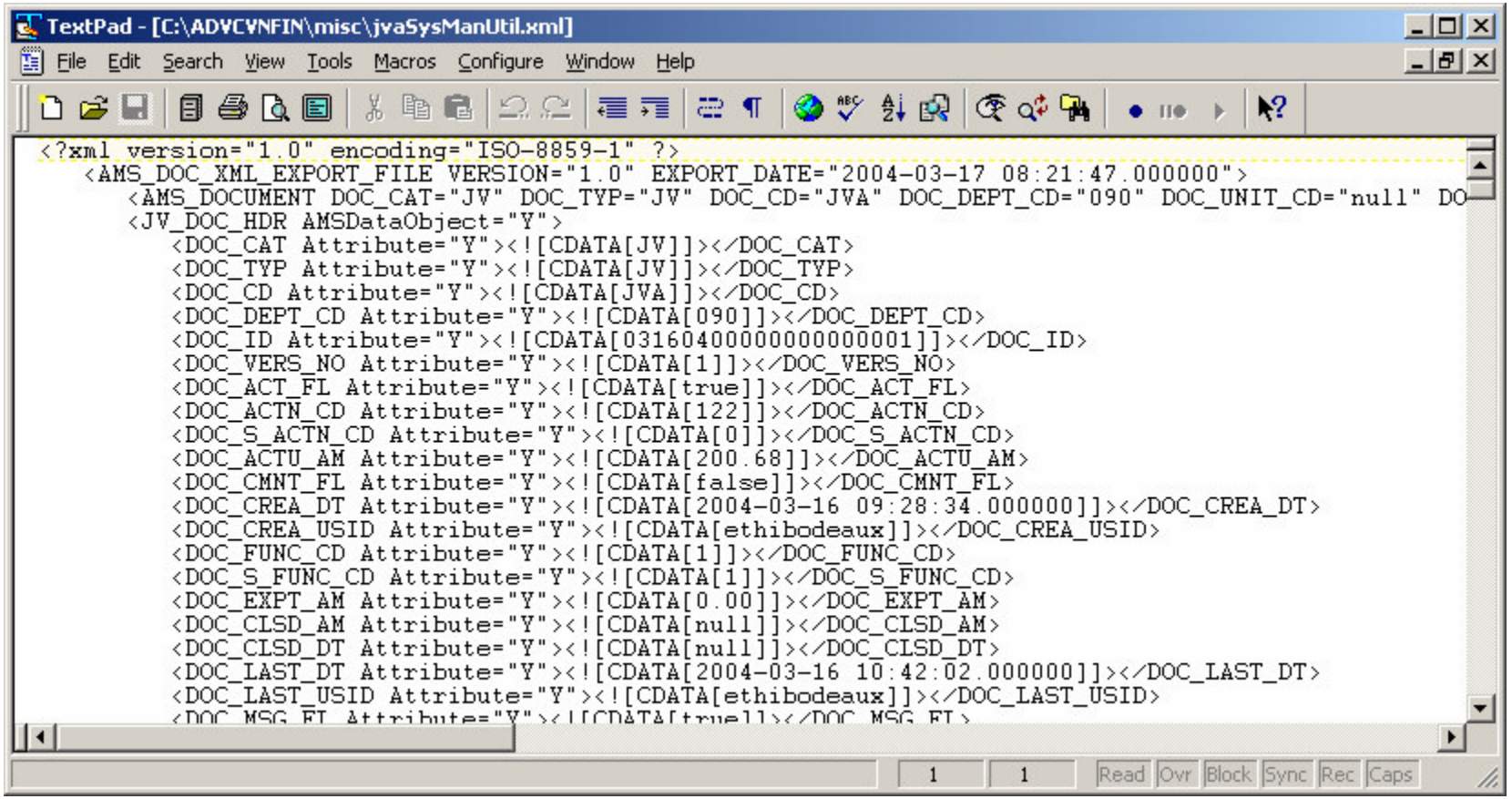
Conversion

- View output XML file



View XML from Sysmanutil

- Look at tags from the XML



Miscellaneous

External Code Module

- Each document conversion should contain a code module for its specific logic
- The name of the code module should have an association to the functional area

Build DOS Batch File

- The DOS Batch file will be used to run all of the necessary items for a certain functional area
- A call to run Conversion Maps to seed the support tables should reside in batch file
- Calls to the infrastructure maps should be called

Send Mail

- Ability to send mail giving process of conversion
- Send e-mail alerts at the conclusion (successful or unsuccessful) of a Map
- External utility

```
*** Execution End: [S_TAB_FQTR_2xbin_3xOrcl] completed successfully
*** Execution Statistics: [S_TAB_FQTR_2xbin_3xOrcl]  executed in 3 minutes, 7.17 seconds
*** Execution Statistics: [S_TAB_FQTR_2xbin_3xOrcl]  Total records read: 29665
*** Execution Statistics: [S_TAB_FQTR_2xbin_3xOrcl]  Total records written: 29179
*** Execution Statistics: [S_TAB_FQTR_2xbin_3xOrcl]  Total reject records written: 0
*** Execution Statistics: [S_TAB_FQTR_2xbin_3xOrcl]  Total records inserted: 29179
*** Execution Statistics: [S_TAB_FQTR_2xbin_3xOrcl]  Total records updated: 0
*** Execution Statistics: [S_TAB_FQTR_2xbin_3xOrcl]  Total records deleted: 0
*** Execution Statistics: [S_TAB_FQTR_2xbin_3xOrcl]  Total records discarded: 486
*** Execution Statistics: [S_TAB_FQTR_2xbin_3xOrcl]  Total records rejected: 0
*** Execution Statistics: [S_TAB_FQTR_2xbin_3xOrcl]  Total error count: 0
Conversion unloaded.
```

Options

- Can run conversion outside DI studio, using the Integration Engine
- Helps performance since the studio is not open
- License allows for both engine and studio to be open

Some .bat files

- Call maps from the dos prompt
  - RDImap.bat
  - StandardInterface_JV_FLEETJV.bat
  - Step1A_transform_all_coa_tables.bat
  - Step2A_transform_all_comm_tables.bat
  - DIBuild_Conv_Inf.bat

Questions